Roberto Brunori Junior

Uma Teoria de Primeira Ordem para Especificação e Análise de Protocolos de Criptografia

Fortaleza

1999

Roberto Brunori Junior

Uma Teoria de Primeira Ordem para Especificação e Análise de Protocolos de Criptografia

Este trabalho foi apresentado à Pós-Graduação em Ciência de Computação do Centro de Ciências da Universidade Federal do Ceará como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciência da Computação.

Orientador: Wamberto Weber M. P. Vasconcelos

Universidade Federal do Ceará
Centro de Ciências
Departamento de Computação
Pós-Graduação em Ciências da Computação

Fortaleza

1999

Esta página foi deixada intencionalmente em branco para numeração correta das páginas quando da inclusão da folha de aprovação.

AGRADECIMENTOS

A Deus, pela saúde e todos os bons momentos de minha vida;

A meus pais, por todo o esforço que têm feito para me proporcionar uma boa criação;

Aos que foram meus professores, pelo esforço em proporcionar a seus alunos uma boa formação;

Em especial, a meu orientador, prof. Wamberto, cujo alto grau de profissionalismo e competência eu até então nunca tinha observado em outros profissionais, pela boa vontade em me atender, mesmo nos momentos mais difíceis;

Ao prof. Fernando Carvalho, por ter incentivado a submissão de meu pedido de aceitação no mestrado;

Aos membros da banca examinadora, por sua valiosa presença e contribuições à melhoria deste trabalho;

À Maxlene Batista, pelo empréstimo da impressora e ajuda na digitação e elaboração das figuras.

RESUMO

Protocolos de criptografia, dentre outras funcionalidades, permitem a comunicação entre agentes em um ambiente computacional de forma a atender os requisitos básicos de segurança, a saber: autenticidade, integridade, confidencialidade e não-repudiação. Diversos protocolos foram propostos e incorporados a tecnologias, incluindo falhas desconhecidas na época de sua aplicação. Tais falhas têm sido descobertas através da aplicação de diferentes métodos formais. Dentre tais métodos, as abordagens lógicas destacam-se por detectarem uma gama maior de falhas.

Neste trabalho, propomos uma teoria de primeira ordem para formalizar a descrição de falhas previamente conhecidas e também verificar a presença de possíveis falhas em protocolos. Os axiomas de nossa teoria representam conhecimento implícito sobre aspectos de um ambiente hostil de comunicação entre agentes. O método de análise resume-se a provas de teoremas, os quais representam possíveis falhas. As provas de teoremas, quando encontradas, representam cenários de ataque, *i.e.*, execuções subvertidas dos protocolos. Especificamos protocolos clássicos e demonstramos formalmente a presença de falhas. A mecanização de provas é possível, conforme demonstramos, apesar das propriedades não-finitárias de nossa teoria.

Palavras-chave: Especificação formal e análise, protocolos de criptografia, lógica de primeira ordem, prova de teoremas.

ABSTRACT

Cryptographic protocols, among other functionalities, allow the communication between agents in a computational environment so that the basic security requirements, *viz.*, authenticity, integrity, confidentiality and non-repudiation, are met. Various protocols have been proposed and incorporated to technologies, including flaws unknown at the age of their application. Such flaws have been discovered through the application of different formal methods. Among these methods, the logical approaches deserve a special place because they are able to detect a larger quantity of flaws.

In this work we propose a first-order theory to formalize the description of previously known flaws as well as to verify possible flaws in protocols. The axioms of our theory represent the implicit knowledge about aspects of a hostile environment for the communication between agents. Our analysis method consists of proving theorems which represent possible flaws. The proof of theorems, when discovered, represent scenarios of an attack, *i.e.*, subverted executions of protocols. We have specified classic protocols and we have formally shown the presence of flaws in them. The automation of proofs is possible, as we have shown, in spite of the non-finitary properties of our theory.

Key words: Formal specification and analysis, cryptographic protocols, first-order logics, theorem proving.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

1 INTRODUÇÃO

Nos últimos anos, o mundo tem acompanhado um rápido desenvolvimento nas áreas dos sistemas computacionais e de telecomunicações. Quase todos os países do mundo estão conectados à Internet, a grande rede de computadores. A indústria de informática nunca vendeu tantos equipamentos. Hoje, sistemas computacionais completos para uso doméstico já podem ser encontrados por menos de US$ 500,00. A telefonia celular chega às mãos até daqueles menos favorecidos economicamente. Satélites e fibras ópticas prometem aumentar a velocidade nas comunicações, permitindo uma nova gama de aplicações. Tecnologias emergentes para comunicação de dados em alta velocidade, como a *Asynchronous Transfer Mode* (ATM), têm experimentado um acelerado desenvolvimento, devido a demanda crescente das aplicações multimídia. O comércio eletrônico já é uma realidade. Algumas grandes empresas já não vendem mais seus produtos da maneira usual, apenas através da Internet. Analistas econômicos apostam no sucesso dessa nova forma de fazer negócio no novo milênio que se aproxima. Neste momento, as tecnologias de segurança na comunicação assumem um papel crucial. Muitas dessas tecnologias já estão incorporadas no dia-a-dia das pessoas, como nas operações bancárias e no comércio eletrônico na Internet. Mas nem tudo é perfeito no mundo das tecnologias de segurança.

Muitas áreas ainda sofrem com problemas relacionados à segurança. Escutas telefônicas, clonagem de telefones celular e fraude no uso de cartões de crédito são notícias comuns, apesar de já existirem tecnologias que provêem soluções para esses problemas. Pior ainda, muitas das tecnologias hoje difundidas e tidas como garantidas, na verdade não oferecem segurança total a seus usuários, podendo ser subvertidas através do agrupamento de recursos computacionais adequados ou simplesmente pela habilidade intelectual de um seleto grupo de pessoas, os *hackers* e *crackers*.

Talvez o problema seja bem maior do que imaginamos, e passe pela alçada dos interesses de governos e grandes grupos econômicos. O fato é que, ao mesmo tempo em que algumas tecnologias têm sucesso reconhecido e vêm ganhando firmeza junto à comu-

nidade de usuários, outras têm sucumbido à descoberta de deficiências que impedem sua funcionalidade. Muitas dessas deficiências encontram-se nas primeiras fases do projeto dos protocolos de criptografia que as suporta, ou seja, não importa quão bom seja o resto do desenvolvimento, uma falha no nível funcional irá sempre se perpetuar até o produto final. É nesse ponto que a aplicação de métodos formais para análise de protocolos de criptografia torna-se útil. Apesar dessa ser uma área ainda nova, já que as primeiras propostas datam de 1989 (BURROWS; ABADI; NEEDHAM, 1990), muitos métodos têm sido usados com sucesso na descoberta de falhas funcionais. Novos métodos têm surgido procurando diminuir a distância conceitual entre a especificação e análise de um protocolo e sua real implementação, como é o caso do proposto por Carlsen (1994b) e da abordagem deste trabalho.

Esta dissertação está inserida no contexto maior do estudo de protocolos de criptografia, suas falhas e dos métodos formais para sua análise. A estrutura geral deste documento é a seguinte.

- No capítulo 2, uma introdução aos conceitos básicos que envolvem os protocolos de criptografia e suas falhas é apresentada. O termo **protocolo** é definido através de exemplo. As principais ameaças à comunicação segura são definidas e relacionadas aos requisitos básicos de segurança. Em seguida, a terminologia e os conceitos básicos de criptografia são apresentados. Conceituamos e apresentamos as funcionalidades da criptografia simétrica, assimétrica, híbrida, funções de *hash one-way* e assinaturas digitais. Exemplos de como esses elementos podem ser combinados para prover funcionalidades mais elaboradas são discutidos. Conceituamos protocolos de criptografia e discutimos diferentes tipos de falhas que esses podem apresentar. As deficiências da maneira usual em que os protocolos são especificados, conhecida como **notação padrão** (CARLSEN, 1994c), são evidenciadas através de exemplos. Analisamos exemplos de falhas clássicas em protocolos para autenticação e distribuição de chaves.

- No capítulo 3, apresentamos uma coletânea de resultados encontrados em referências bibliográficas sobre métodos formais para análise de protocolos de criptografia. Descrevemos também as principais características e escopo de cada método. Outros trabalhos relacionados são também comentados.

- No capítulo 4, apresentamos uma proposta de método formal de análise de protocolos de criptografia. Nosso objetivo é prover os analistas de um método para formalizar a descrição de protocolos e suas falhas já conhecidas, bem como para

verificar a presença de possíveis falhas. O esquema geral do método é apresentado, tratando-se basicamente do uso de uma linguagem formal e um sistema de axiomas para prova de teoremas. Destacamos pontos divergentes e de suma importância a respeito da especificação e análise de protocolos. Nossos comentários sobre tais pontos serviram como princípios de projeto para a abordagem proposta. Definimos a sintaxe e semântica de uma linguagem de primeira ordem para especificar protocolos. Axiomatizamos o conhecimento tácito sobre o ambiente que envolve os protocolos e seus elementos. Na tentativa de representar de forma realista tal ambiente, ambos os tipos de características, boas e más, são modeladas. Em seguida, tecemos comentários sobre como usar a linguagem proposta para especificar os pontos importantes anteriormente discutidos e apresentamos exemplos práticos de trechos de especificação. Estendemos a linguagem utilizada a fim de também representarmos fórmulas que representam possíveis falhas. Exemplos de representação de falhas são mostrados.

- No capítulo 5, empregamos a abordagem sugerida no capítulo anterior na análise de 3 protocolos clássicos, sendo dois de autenticação e distribuição de falhas e um para simples transferência de informação com confidencialidade e sem autenticação. Com isso, ilustramos as capacidades dessa abordagem, inclusive para analisar classes distintas de protocolos. Os protocolos apresentados possuem diversas falhas de vários tipos. Tais falhas são bem conhecidas e são bastante discutidas na literatura sobre o assunto. Iniciamos reformulando a estrutura de aplicação do método proposto de acordo com as definições do capítulo anterior. Dada a inviabilidade de apresentar demonstrações em sua forma original (árvores de prova do cálculo de seqüentes), exemplificamos o uso de um algoritmo simples para **aplanagem** de árvores de prova, descrito no apêndice B. Após apresentarmos as demonstrações das falhas nos protocolos, realizamos uma comparação de resultados com outros métodos e discutimos algumas propriedades da abordagem proposta.

- No capítulo 6, descrevemos nossa experiência para automatizar a aplicação do método de análise de protocolos apresentado nos capítulos anteriores. Propomos uma linguagem de programação em lógica de propósito específico. Tal linguagem é um subconjunto da linguagem PROLOG. Mostramos como as sentenças da linguagem proposta no capítulo 4 podem ser facilmente mapeadas para sentenças dessa linguagem. Alguns axiomas da teoria proposta são então reescritos sob a forma de cláusulas dessa linguagem de programação. Para que pudéssemos usar o mecanismo de prova existente nos interpretadores PROLOG, dois obstáculos tiveram de

ser vencidos. O primeiro diz respeito as propriedades não finitárias do modelo de computação proposto no capítulo 4. O segundo está relacionado com a possibilidade do processo de prova de teoremas do PROLOG entrar num laço que nunca acaba, mesmo quando existe uma prova. Parte do primeiro problema foi resolvido usando-se limites para o domínio dos números naturais, presente em nosso modelo. Para resolver a parte restante e o segundo problema simultaneamente, propomos o uso da técnica de busca com aprofundamento iterativo. Desenvolvemos um meta-interpretador como implementação dessa técnica. Finalmente, um exemplo completo, o do protocolo de Nesset (NESSET, 1990), é especificado e uma tentativa de análise automatizada é descrita.

- No capítulo 7, tecemos comentários sobre os resultados obtidos, lições aprendidas e trabalhos futuros.

- No apêndice A, apresentamos as provas completas dos teoremas do capítulo 5, mostradas anteriormente de forma resumida.

- O apêndice B contém a descrição das regras do cálculo de seqüentes utilizadas neste trabalho e a descrição do algoritmo de aplanagem de árvores de prova citado anteriormente.

- O apêndice C contém o código completo da tentativa de análise automatizada do protocolo de Nesset.

2 PROTOCOLOS DE CRIPTOGRAFIA E SUAS FALHAS

Neste capítulo, apresentamos uma introdução aos conceitos básicos envolvendo protocolos de criptografia e suas falhas. Iniciamos com a terminologia utilizada no restante do trabalho. Apresentamos os elementos básicos da criptografia e suas funcionalidades. O conceito de protocolo de criptografia é então apresentado com exemplos. A maneira usual como protocolos de criptografia são apresentados, batizada por Carlsen (1994c) como **notação padrão**, é apresentada. As deficiências da notação padrão são discutidas através de exemplos. Segue-se uma discussão sobre falhas em protocolos de criptografia para autenticação e distribuição de chaves.

Nosso intuito não é exaurir o assunto, e sim prover o leitor da intuição sobre a funcionalidade dos protocolos de criptografia e os problemas que surgem devido a seu uso. Como outras introduções ao assunto e leitura aprofundada, sugerimos Schneier (1996), Simons (1996), Tanenbaum (1996), Stallings (1995), Hughes (1995), Clark e Jacob (1996) e Oppliger (1996).

2.1 Protocolos

Protocolos são conjuntos de regras, convenções e ações que devem ser seguidas por dois ou mais **agentes** para realização de uma tarefa. O termo "agente", no contexto aqui apresentado, é uma generalização para qualquer entidade, seja ela um ser humano, um computador ou processo, que esteja envolvido na realização de uma tarefa. Um protocolo envolve necessariamente duas ou mais partes. As ações descritas num protocolo seguem uma ordenação lógica e geralmente possuem uma interdependência, impedindo que o início de uma ação aconteça antes do fim de outra. **Protocolos de comunicação** definem as ações e regras que efetivam a comunicação entre agentes. O uso efetivo de protocolos

pressupõe ainda os seguintes aspectos:

- Todos os agentes envolvidos conhecem o protocolo a ser usado e concordam quanto a seu uso;
- Cada passo do protocolo está bem definido, não deixando margens a diferentes interpretações por parte dos agentes;
- Há sempre uma ação especificada para cada possível estado em que os agentes venham a se encontrar;
- Se qualquer dos passos do protocolo não puder ser completado com sucesso, o agente envolvido deve abortar o protocolo, voltando a seu estado inicial.

Como exemplo de uma tarefa governada por um protocolo informal simples, podemos citar o atendimento automático num caixa eletrônico. Para que, por exemplo, uma consulta aos dados de uma conta possa ser realizada, os seguintes passos devem ocorrer.

1. O cliente aperta o botão de saldo ou extrato, de acordo com sua escolha.
2. O caixa eletrônico solicita que o cliente passe seu cartão magnético pela máquina.
3. O cliente passa o cartão, o qual contém os dados que o identificam.
4. O caixa eletrônico solicita que o cliente digite sua senha secreta.
5. O cliente digita sua senha.
6. O caixa eletrônico verifica se a senha digitada está completa e imprime os dados solicitados.

Deve-se notar a interdependência entre as ações. Se um dos passos não for completado com sucesso, ou se a seqüência das ações for mudada, o protocolo irá falhar, e a tarefa não será realizada. Esse é um exemplo simples, onde foram descritas, informalmente, apenas as ações a serem realizadas pelos agentes. Foram omitidas as regras que completam a definição do protocolo. Nosso objetivo aqui é apenas familiarizar o leitor com alguns conceitos intuitivos, e não fornecer definições formais para os elementos de um protocolo.

Antes de prosseguirmos com a discussão sobre protocolos, vamos conhecer as principais ameaças envolvidas na comunicação.

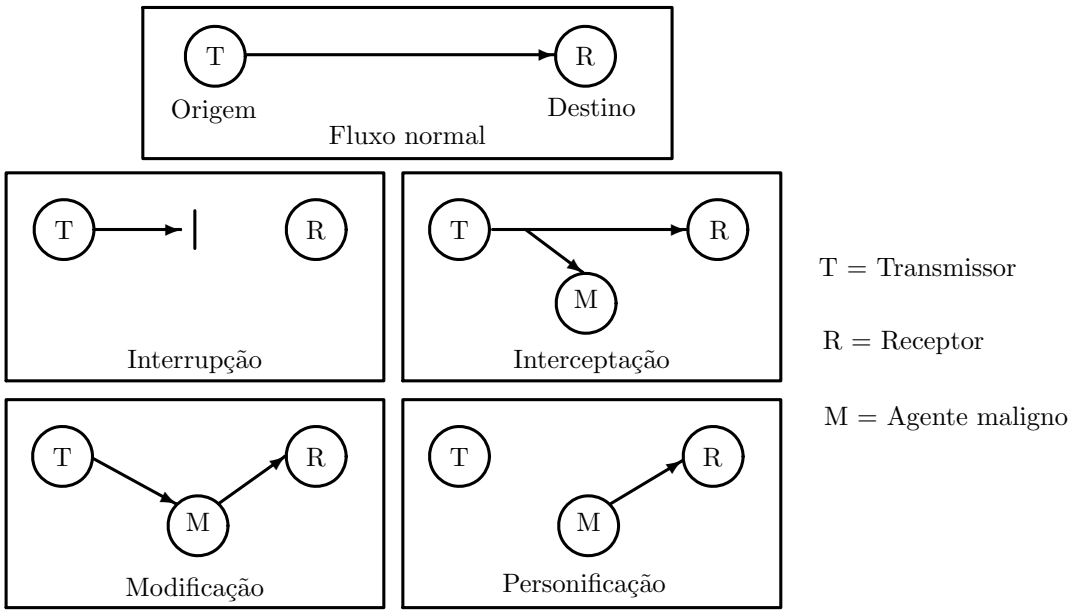

Figura 1: Ameaças à comunicação segura.

2.2 Ameaças à Comunicação Segura

Na comunicação entre agentes, identificamos 4 classes básicas de ameaças à segurança (CARLSEN, 1994b, cap. 1), como mostra a figura 1.

Os ataques de **interrupção** fazem parte de uma classe maior de ameaças conhecida como **negação de Serviços**[1], onde um agente fica impossibilitado de utilizar um recurso, o qual normalmente ele teria acesso. No caso particular da interrupção, um agente fica sem comunicação. Esses ataques são difíceis de serem evitados através do uso de protocolos, pois geralmente estão relacionados ao acesso físico aos recursos, o que dificilmente é representado nos protocolos.

Num ataque de **interceptação**, um terceiro agente, que não o transmissor ou receptor, consegue ter acesso ao conteúdo da mensagem transmitida. Os ataques de interceptação são facilmente concebíveis, uma vez que os canais de comunicação são geralmente compartilhados por muitos agentes.

Na **modificação**, a mensagem sendo transmitida tem seu conteúdo alterado. A modificação é um ataque mais difícil de ser concebido que a interceptação, pois o agente maligno geralmente precisa impedir que a mensagem original chegue a seu destino, sob pena de o ataque não ser bem sucedido. Num ambiente de inter-redes, esse ataque é mais provável, uma vez que a passagem da mensagem de uma rede para outra é facilmente

[1]Do inglês *Denial of Service*, ou DoS.

evitada, podendo então a mensagem ser substituída e daí passada adiante.

Na **personificação**, um agente faz-se passar por outro, na tentativa de obter recursos que normalmente não lhe seriam fornecidos. A personificação pode ser considerada como um caso particular da modificação, pois para sua concepção, basta que o agente maligno substitua a identidade do agente transmissor, contida na mensagem, por sua própria identidade.

Segundo Carlsen (1994b, cap. 1), quatro requisitos básicos caracterizam a comunicação segura.

Autenticação – Um agente deve ter certeza de que está se comunicando com o agente correto, e não com um outro que tenta passar-se por esse. Em algumas aplicações, é desejável que a autenticação seja mútua.

Integridade – Uma mensagem enviada deve chegar a seu destino como foi criada, ou seja, sem alterações.

Confidencialidade – Somente o(s) destinatário(s) original(is) devem ter acesso ao conteúdo de uma mensagem transmitida.

Não-repudiação – Um agente não pode negar o fato de ter criado ou enviado uma mensagem.

Voltando à figura 1, podemos observar quais requisitos não foram atendidos em cada uma das ameaças. Na interceptação, não há confidencialidade. Na modificação, não há confidencialidade nem integridade. Na personificação, falta autenticidade. **Protocolos de segurança** são aqueles criados com o objetivo de prover os requisitos acima na comunicação entre agentes. Deste ponto em diante, quando citarmos o termo **protocolo**, estaremos nos referindo aos protocolos de segurança. A grande problemática no projeto de protocolos que tratam do problema de segurança na comunicação é garantir, em parte ou completamente, esses requisitos, simultaneamente.

2.3 Conceitos Básicos de Criptografia

Vamos seguir a mesma terminologia adotada por Schneier (1996, p. 1). O processo de transformação de uma mensagem de modo a esconder o seu significado é chamado **criptografar** (ou encriptar). Uma mensagem que sofre criptografia também é conhecida

Figura 2: Relação entre os termos básicos de criptografia.

como **criptotexto**. Uma mensagem não transformada pode ser tratada como **texto aberto** ou **texto simples**. O processo de transformar uma mensagem criptografada de volta numa mensagem com significado é conhecido como **descriptografar** (ou decriptar). Esses termos estão relacionados de acordo com a figura 2.

A arte e ciência de manter mensagens seguras é conhecida como **criptografia** é praticada por **criptógrafos**. **Criptoanalistas** são praticantes da **criptoanálise**, a arte e ciência de subverter o processo de criptografia, isto é, conhecer o significado de uma mensagem, mesmo estando essa criptografada. O ramo da matemática que envolve tanto a criptografia quanto a criptoanálise é a **criptologia** e é estudado por **criptologistas**.

A fim de atender os requisitos de segurança, mensagens trocadas em protocolos são manipuladas através de **funções de criptografia**. Vamos conhecer como essas funções são utilizadas e que tipo de funcionalidade elas são capazes de prover. Vamos considerar essas funções como **modelos de criptografia**, ou seja, vamos tratá-las como "caixas-pretas" e abstrair-nos de seu processamento interno, preocupando-se apenas com suas entradas e saídas. Finalmente, vamos agrupá-las e observar como seu uso pode levar os protocolos a atenderem os requisitos de segurança.

2.3.1 Criptografia Simétrica

Neste modelo, uma mensagem (entrada) é criptografada usando-se uma função parametrizada por uma **chave**, ou seja, uma seqüência qualquer de bits que deve ser mantida em segredo por um grupo seleto de agentes que desejam comunicar-se com confidencialidade. A função inversa, ou seja, a descriptografia, deve utilizar como parâmetro a mesma chave utilizada na encriptação, como mostra a figura 3. A chave só deve ser conhecida pelos agentes que estão buscando privacidade na comunicação. Qualquer outro agente que venha a obter a chave poderá descriptografar as mensagens transmitidas, uma vez que essas tenham sido interceptadas. A principal dificuldade na utilização desse modelo está na distribuição da chave secreta entre os agentes que querem comunicar-se com segurança. É importante garantir que a chave seja distribuída com autenticidade e confidencialidade

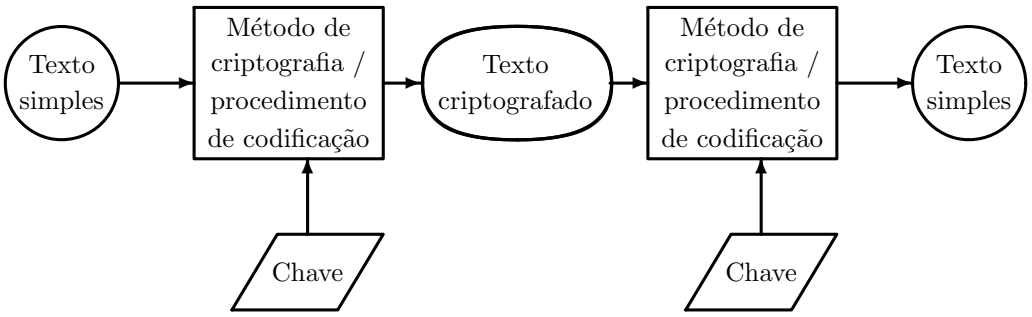

Figura 3: Criptografia simétrica.

entre os agentes corretos, antes que esses possam usar de fato a criptografia simétrica para prover comunicação segura. Deve-se notar o caráter circular do problema da distribuição de chaves do modelo simétrico, pois, para garantir autenticidade e confidencialidade, é preciso utilizar um canal de comunicação já considerado seguro. Na prática, o uso da criptografia simétrica como único meio de prover confidencialidade é feito através da entrega direta da chave de um agente para outro, ou seja, buscando eliminar o uso de canais compartilhados.

É importante frisar que as funções de criptografia simétrica não precisam ser secretas para que forneçam a funcionalidade desejada. De fato, seu comportamento é geralmente bem conhecido por todos os agentes envolvidos na comunicação, inclusive pelos agentes malignos. Toda a segurança desse modelo está baseada no conhecimento da chave secreta.

Quando afirmamos que uma mensagem criptografada não pode ser descriptografada sem o conhecimento da chave secreta, não implica dizer que não haja outros meios de se obter a mensagem original a partir do criptotexto (mensagem criptografada). Podemos, por exemplo, conceber um ataque onde são tentadas todas as combinações possíveis de chave secreta. Esse ataque é conhecido como **força bruta**. Os bons algoritmos de criptografia têm a propriedade de que é computacionalmente inviável (*i.e.*, exige uma capacidade de processamento ou armazenamento indisponível) obter a chave através de ataques de força bruta, ou através de métodos de criptoanálise. O mais conhecido algoritmo de criptografia simétrica é o *Data Encryption Standard*, ou DES, que usa chaves de 56 bits e pode operar em vários modos: por blocos de dados, com auto alimentação (saída de um passo servindo de parâmetro para o próximo passo) ou ainda com fluxos de bits como entrada. Detalhes sobre o DES, bem como uma variedade de outros algoritmos de criptografia simétricos, são descritos em Schneier (1996) e Kaufman, Perlman e Speciner (1995).

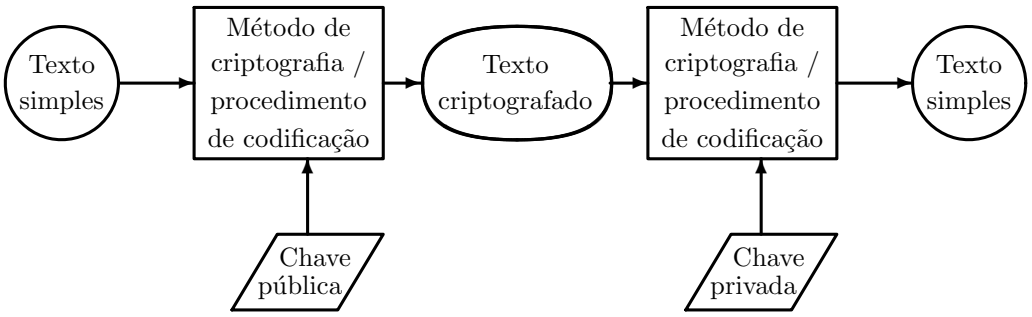

Figura 4: Criptografia assimétrica.

2.3.2 Criptografia de Chave Pública

Neste modelo, cada agente deve possuir um par de chaves: a **chave privada** e a **chave pública**. Esse par de chaves possui propriedades matemáticas que permitem que as funções de criptografia de chave pública possuam a seguinte característica fundamental: o que é criptografado com a chave pública só pode ser descriptografado com a chave privada correspondente. Daí o outro nome desse modelo: **criptografia assimétrica**. Esse esquema de funcionamento é mostrado na figura 4. Em alguns algoritmos de criptografia assimétrica, como o RSA [2], as mensagens criptografadas com uma chave privada só podem ser descriptografadas com a chave pública correspondente. A primeira das características citadas pode ser usada para garantir confidencialidade e a segunda para garantir autenticidade (através do uso de assinaturas digitais, que abordaremos mais adiante). A distribuição de chaves fica facilitada, pois agora basta que a chave pública seja distribuída com autenticidade, já que todos os agentes, inclusive os malignos, podem conhecer as chaves públicas uns dos outros, sem prejuízo para a confidencialidade. O maior problema na utilização desse modelo está em garantir que uma chave pública seja realmente de quem parece ser, e não de outro agente que pode estar tentando uma personificação. Para ilustrar a importância dessa garantia, vamos discutir o ataque clássico aos sistemas de criptografia assimétricos conhecido como *man-in-the-middle* (SCHNEIER, 1996, p. 48).

Vamos supor que dois agentes quaisquer, A e B, cada um deles possuidores de um par de chaves do modelo assimétrico, desejam comunicar-se com confidencialidade. Supondo ainda a presença de um agente maligno M, também possuidor de um par de chaves. A seguinte seqüência de eventos, esquematizada na figura 5, evidencia uma falha de segurança na comunicação entre A e B.

[2] Sigla que vem do nome de seus criadores.

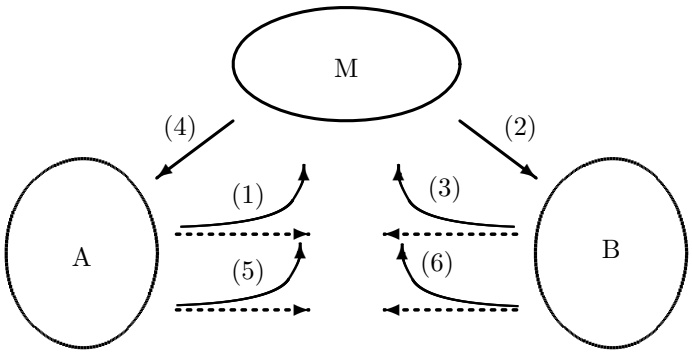

Figura 5: Ataque *Man-in-the-Middle*. As setas contínuas indicam o fluxo real das mensagens.

1. O agente A envia para o agente B sua chave pública, aqui denotada por K_A.

2. O agente maligno M intercepta a chave K_A e envia para B sua própria chave pública, K_M.

3. Por não contar com a autenticidade, o agente B recebe K_M como sendo a chave pública de A. O agente B envia para A sua chave pública, K_B.

4. A chave pública de B também é interceptada por M e substituída por K_M. Da mesma forma que B, A é enganado, e levado a concluir que K_M é a chave pública de B.

5. Quando A envia uma mensagem criptografada com a suposta chave pública de B, o agente M a intercepta. Uma vez que a mensagem está na realidade criptografada com sua própria chave pública (e não a de B), o agente M descriptografa a mensagem, tendo assim acesso a seu conteúdo. Já que a chave pública de B foi anteriormente capturada por M, esse agente pode criptografar novamente a mensagem com K_B e enviá-la para B.

6. De forma análoga ao passo anterior, quando B envia uma mensagem para A, essa é interceptada por M, descriptografada com K_M, criptografada com K_A e enviada para A.

Note que os agentes A e B vão considerar sua comunicação como confidencial, o que claramente não é verdade. Essa falha é devida basicamente à falta de autenticidade na distribuição das chaves públicas.

Certificados de chave pública são uma maneira de unir a chave pública de um agente à sua identidade. Falaremos mais sobre certificados depois de conhecermos o conceito de assinatura digital.

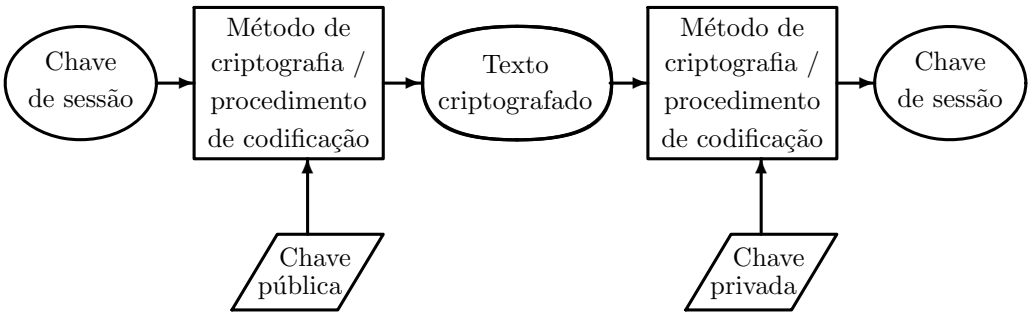

Figura 6: Modelo Híbrido.

2.3.3 Modelo Híbrido

A criptografia de chave pública demanda muito tempo de processamento. Se todas as mensagens a serem transmitidas tivessem de passar por um algoritmo de criptografia assimétrico, teríamos um atraso que pode inviabilizar a comunicação (SCHNEIER, 1996, p. 33). Para evitar esse problema, alguns protocolos são projetados com o objetivo de distribuir uma chave secreta (usada no modelo simétrico) entre os agentes comunicantes usando o modelo assimétrico (ver fig. 6). Uma vez realizados os passos do protocolo, que em geral não demandam muito tempo, e que tenhamos distribuído uma chave secreta com autenticidade e confidencialidade, podemos passar a usar o modelo simétrico durante o resto da sessão de comunicação entre os agentes. Esse modelo tem ainda a vantagem de a chave secreta só ser usada durante uma sessão, dificultando a criptoanálise das mensagens, já que não vai existir uma grande quantidade de mensagens criptografadas com a mesma chave. Mesmo que a criptoanálise obtivesse sucesso e a chave usada numa sessão passada tenha sido descoberta, sessões futuras não estariam necessariamente comprometidas.

2.3.4 Funções de *Hash One-Way*

Algumas funções possuem a propriedade de que é fácil (*i.e.*, é computacionalmente factível) calcular seu valor dada uma entrada qualquer, mas é difícil calcular o valor da entrada inicial dado o valor da função correspondente, ou seja, é difícil achar a sua função inversa. Essas funções são conhecidas como **funções *one-way***. Se, além disso, essas funções tiverem a propriedade de que, dada uma entrada de tamanho arbitrário, o valor correspondente da função tiver tamanho fixo, então estaremos diante de uma função de *hash one-way* (ver fig. 7). O resultado da aplicação de uma função de *hash one-way* é comumente conhecido como ***message digest***. Uma outra propriedade desejável nessas

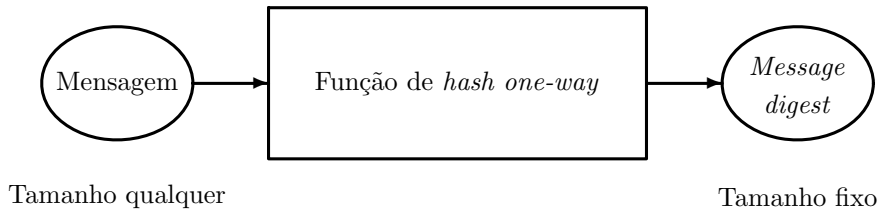

Figura 7: Função de *Hash One-Way.*

funções é a de que a probabilidade de encontrar duas entradas que resultem no mesmo valor de função é pequena. Funções de *hash* com essa propriedade são classificadas como **livres de colisão**. Uma aplicação desse tipo de função encontra-se na autenticação de usuários em sistemas de computação. O sistema teria armazenado não uma chave secreta (senha) compartilhada com o usuário, mas sim o valor de uma função *one-way* dessa senha. Essa técnica é usada para proteger as senhas dos usuários no caso de invasão do sistema. No momento da autenticação, o usuário entra com a sua senha, o sistema calcula o valor da função e compara com o valor armazenado, se os valores forem os mesmos, o usuário está autenticado. Esta é uma forma básica de autenticação. Na prática, são aplicadas outras técnicas, como o uso de *salt* e do algoritmo de criptografia simétrica dos sistemas UNIX. Mais detalhes podem ser encontradas em Garfinkel e Spafford (1996) e Kaufman, Perlman e Speciner (1995).

2.3.5 Assinaturas Digitais

Uma assinatura digital em um documento eletrônico tem o mesmo propósito de uma assinatura comum em um documento em papel, isto é, provar a identidade do criador do documento. Assinaturas digitais podem ser criadas usando-se os modelos simétricos e assimétricos, em conjunção com funções de *hash*. A maneira usual acontece como descrito abaixo (ver fig. 8). Calculamos o valor de uma função de *hash one-way* da mensagem e criptografamos esse valor com a chave privada do modelo assimétrico, criando assim uma **assinatura digital**. A assinatura digital depende da mensagem usada para gerá-la e pode ser verificada por qualquer agente que conheça a chave pública correspondente à chave privada que foi usada na criação da assinatura. As assinaturas digitais não têm como objetivo prover confidencialidade, e sim autenticidade e integridade. A mensagem é transmitida em texto aberto, juntamente com sua assinatura. Para verificar a assinatura, o receptor deve descriptografá-la com a chave pública do agente que afirma ter assinado a mensagem, calcular a função de *hash* da mensagem assinada e comparar os dois valores.

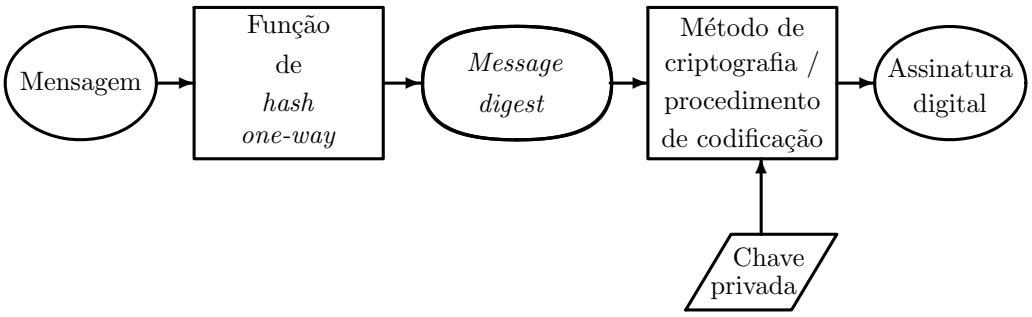

Figura 8: Assinatura digital.

Se os valores forem os mesmos, o receptor tem garantias de que a mensagem foi mesmo assinada pelo agente que corresponde à chave pública utilizada na descriptografia da assinatura e que a mensagem não foi alterada durante a transmissão, já que o valor da função de *hash* transmitido (na assinatura) e do calculado (a partir da mensagem) são os mesmos. Essa forma de assinatura digital possui ainda a vantagem de não acrescentar uma grande sobrecarga na comunicação, uma vez que a assinatura tem um tamanho fixo e gerealmente menor que o da mensagem.

2.3.6 Certificados de Chave Pública

Assinaturas digitais podem ser usadas para criar o que chamamos de **certificado de chave pública**. Um certificado consiste na identidade do agente, sua chave pública e alguns dados adicionais, como o período de validade do certificado, tudo isso assinado por um agente que deve ser confiável aos agentes que desejam comunicar-se. O uso de certificados facilita a distribuição de chaves públicas com autenticidade, já que agora um agente pode, ao receber um certificado, verificar sua autenticidade, desde que esse agente conheça a real chave pública do agente que emitiu o certificado. Essa chave pública deve ter sido previamente distribuída com autenticidade e integridade. Certificados podem também ser postos em serviços de diretório, de forma que qualquer agente possa obter o certificado de qualquer outro. Para verificar a autenticidade de um certificado, é preciso que o agente verificador possua a chave pública do agente que emitiu o certificado, o que nem sempre acontece. Contudo, esse agente pode possuir a chave pública de algum outro agente que por sua vez pode possuir a chave pública desejada e repassá-la para o primeiro agente, formando assim uma **cadeia de certificação**. O importante é que o primeiro agente receba, com autenticidade, a chave pública do agente que emitiu o certificado.

2.4 Protocolos de Criptografia

Protocolos de criptografia são aqueles que usam os elementos da criptografia (funções, chaves, certificados etc.) como parte de suas ações e regras. Protocolos de criptografia são concebidos para resolver problemas relacionados à confidencialidade, integridade, autenticidade e não-repudiação. Como exemplo, vamos descrever, em linguagem natural, um problema que envolve os requisitos de segurança e um protocolo simples capaz de solucioná-lo (SIMONS, 1996).

Suponha que dois amigos estão conversando ao telefone e decidem ir ao cinema juntos. Contudo, eles não entram em acordo a respeito de qual filme assistir. Eles resolvem então deixar que a sorte decida por eles, e concordam em usar o conhecido "cara-ou-coroa" para resolver a questão. Se eles estivessem frente-a-frente, não haveria maiores problemas, mas como jogar o cara-ou-coroa se os amigos estão conversando por telefone? O amigo que jogasse a moeda poderia mentir a respeito do resultado obtido, e o outro amigo não teria como confirmar o resultado. No entanto, os amigos poderiam usar o seguinte protocolo.

1. Os amigos entram em acordo e decidem usar uma determinada função *one-way*, conhecida por ambos, e associam os lados da moeda a dois números quaisquer distintos.

2. Um dos amigos joga a moeda e concatena o número associado ao resultado a um outro número qualquer aleatório, depois calcula o valor da função *one-way* aplicada ao resultado da concatenação e envia o resultado final para o segundo amigo.

3. O segundo amigo é convidado a escolher cara ou coroa.

4. De acordo com a escolha, o primeiro amigo anuncia sua derrota ou vitória e envia para o segundo amigo o número aleatório usado anteriormente, juntamente com o suposto resultado da jogada da moeda.

5. O segundo amigo então pode conferir sua derrota ou vitória, bastando para isso realizar a mesma operação de concatenação e aplicação da função *one-way* sobre os dados recém recebidos. Se os valores finais forem os mesmos, o segundo amigo pode ter certeza que o primeiro amigo dizia a verdade quando anunciou o resultado.

Esse protocolo funciona, pois o segundo amigo não tem como deduzir o resultado da jogada da moeda possuindo somente o resultado da aplicação da função *one-way*. Se o primeiro amigo tentar mentir sobre o resultado da jogada da moeda, o cálculo realizado

no último passo pelo segundo amigo não corresponderá ao valor recebido anteriormente. Contudo, uma vez que tenha descoberto sua derrota, nada impede o primeiro amigo de abortar a execução do protocolo no início do passo 4, e assim não enviar o número aleatório usado anteriormente para que o segundo amigo possa conferir o resultado. Dependendo do que tiver sido acertado como procedimento de recuperação de erro, ao longo de várias tentativas, o primeiro amigo pode passar a ter uma probabilidade próxima de 1 de vencer o desafio. Tal falha é descrita e classificada por Carlsen (1994b, p. 28) como **falha de repudiação**.

Outro exemplo mais elaborado é como os programas navegadores da Internet podem garantir que os dados enviados através da rede só sejam acessíveis ao servidor com o qual a conexão estiver estabelecida. Vamos tratar os agentes como **cliente** e **servidor**, em menção ao modelo de comunicação utilizado nas aplicações Internet. O protocolo a seguir tem como objetivo básico o estabelecimento de uma chave de sessão entre o cliente e o servidor, a ser utilizada no modelo de criptografia simétrico durante o restante da conexão. Mais uma vez, apresentaremos apenas uma descrição de alto nível do funcionamento do protocolo, detalhes sendo encontrados em Kocher, Freier e Karlton (1996).

1. O cliente envia para o servidor uma mensagem que sugere quais algoritmos devem ser usados no restante do protocolo. No caso do servidor não aceitar esses parâmetros, uma mensagem de erro é enviada de volta ao cliente.

2. O servidor envia seu certificado de chave pública assinado por um agente confiável ao cliente. Opcionalmente, o servidor pode solicitar que lhe seja enviado o certificado de chave pública do cliente, de acordo com o que estiver sido estabelecido no primeiro passo do protocolo.

3. O cliente verifica a autenticidade e validade do certificado de chave pública do servidor, usando a chave pública do agente que emitiu o certificado do servidor. Essa chave deve ter sido previamente distribuída com autenticidade.

4. O cliente gera uma chave de sessão, criptografa essa chave com a chave pública do servidor e a envia, juntamente com seu certificado de chave pública (se for o caso).

O servidor e o cliente têm agora uma chave de sessão estabelecida com autenticidade e confidencialidade, e podem usar o modelo de criptografia simétrico para criptografar dados durante o restante da conexão. A autenticação pode ser unilateral (do servidor

apenas) pois, geralmente, o único a fornecer dados sensíveis é o cliente. No caso do servidor também ter que fornecer dados sensíveis para o cliente, o envio do certificado do cliente torna-se obrigatório. Neste ponto, poderia surgir a pergunta de como são distribuídos os certificados de chave pública dos agentes confiáveis. No caso dos navegadores da Internet, esse problema foi resolvido simplesmente distribuindo-se o proprio navegador já com os certificados das chamadas **autoridades de certificação**. Autoridades de certificação são entidades responsáveis pelo cadastro e emissão dos certificados de chave pública. Essas entidades devem ser de idoneidade reconhecida por todos os agentes participantes. Os agentes interessados em ter suas chaves públicas assinadas por uma autoridade de certificação devem solicitar diretamente tal serviço, diminuindo ao máximo a possibilidade de falta de autenticação na distribuição do certificado. A cada nova versão dos navegadores, novos certificados são adicionados. Quando um navegador receber um certificado de um servidor, ele será capaz de verificar a autenticidade desse certificado se ele possuir o certificado, e portanto a chave pública, da entidade que emitiu o certificado recebido. O servidor, por sua vez, deve estar cadastrado junto a alguma autoridade de certificação, onde teve seu certificado emitido.

Tal solução só funciona na prática devido a grande popularidade e pequeno número de navegadores disponíveis no mercado. O protocolo e a solução de distribuição de certificados descritos acima de nada adiantam no caso de ataques com **cavalos de troia** (HOFFMAN, 1990). Nada impede que seja criado um navegador que se pareça exatamente como algum dos mais consagrados e se comporte aparentemente exatamente como esse, mas que, ao invés seguir o protocolo descrito acima, envia os dados do usuário pela rede sem proteção e faz parecer que uma conexão segura está presente.

2.5 Notação Padrão

Protocolos de criptografia são usualmente descritos com o que Carlsen (1994c) chama de **notação padrão**. O termo "padrão" parece ser mal empregado, uma vez que na literatura são encontradas diversas variações para a sintaxe dessa notação. Contudo, as diferenças na sintaxe não alteram o significado básico das especificações, as quais procuram indicar o real conteúdo das mensagens trocadas entre os agentes. Neste trabalho, as seguintes convenções a respeito da notação padrão serão adotadas.

- A, B, C, S – Esses símbolos representam os agentes que desejam comunicar-se atendendo os requisitos de segurança. O símbolo $\boldsymbol{S}$ denota o agente responsável pela

geração e distribuição das chaves de sessão na grande maioria dos protocolos de autenticação e distribuição de chaves. O símbolo $\boldsymbol{A}$ é o agente **iniciador**, ou seja, aquele que toma a iniciativa para o estabelecimento de uma chave de sessão. O **respondedor**, representado por $\boldsymbol{B}$, é o agente com o qual o iniciador deseja estabelecer uma chave. Quando esse tipo de símbolo aparecer como uma mensagem, devemos entendê-lo como a **identidade** do agente correspondente. Geralmente, é assumido que tanto o iniciador quanto o respondedor já compartilham chaves com o agente S antes do início de qualquer execução do protocolo. Os protocolos não tratam da questão de como essas chaves compartilhadas foram inicialmente estabelecidas, mas se valem dessa suposição para atingir seus objetivos. O símbolo $\boldsymbol{C}$ representa um agente maligno, o qual tenta subverter o protocolo, de forma a obter alguma mensagem que deveria ser confidencial a A e B somente (talvez a própria chave de sessão a qual está sendo estabelecida), ou levar o iniciador ou o respondedor a reutilizar uma chave de sessão, possivelmente comprometida.

- $N_a, N_b, \ldots$ – A expressão $\boldsymbol{N_a}$ representa um ***nonce*** gerado pelo agente A. *Nonces* são valores numéricos gerados aleatoriamente. Seu intuito é o de servir como objeto de um desafio, onde um agente envia um *nonce* para outro agente e espera receber uma resposta que contenha, dentre outras mensagens, um *nonce* com o mesmo valor que o anteriormente enviado. Desta forma, o agente tem garantias que a mensagem recebida foi gerada na atual execução do protocolo, e não numa execução passada. Essa garantia é de suma importância no caso da mensagem recebida ser a chave de sessão. Geralmente, o *nonce* enviado não precisa estar criptografado, mas a mensagem de resposta (que pode ser a chave de sessão) deve estar, juntamente com o *nonce*, criptografada com uma chave conhecida pelo agente.

- $K_{bs}, K_{as}, \ldots$ – O termo $\boldsymbol{K_{as}}$ denota a chave compartilhada entre os agentes A e S. Essas são chaves do modelo simétrico de criptografia (ver 2.3.1), ou seja, uma mensagem criptografada com K_{bs}, por exemplo, só pode ser descriptografada por agentes conhecedores da chave K_{bs}, dentre os quais, possivelmente, podem encontrar-se outros agentes que não B e S. As chaves de sessão entre o iniciador e o respondedor são denotadas por K_{ab}.

- $K_a, K_b, \ldots, K_a^{-1}, K_b^{-1}, \ldots$ – A expressão $\boldsymbol{K_a}$ representa a chave pública do agente A, já $\boldsymbol{K_a^{-1}}$ é a chave privada correspondente. Obviamente, essas são chaves do modelo assimétrico de criptografia (ver 2.3.2).

- $\{M\}K_{as}, \{M\}K_b, \ldots$ – Os símbolos **{** e **}** são usados para indicar as funções de

criptografia simétrica e assimétrica. A expressão $\{M\}K_{as}$ indica uma mensagem M criptografada com a chave K_{as}. O fato de a chave usada ser a do modelo simétrico indica que a função de criptografia é simétrica. No caso da chave usada ser do tipo K_a, fica convencionado que uma função de criptografia assimétrica está sendo usada. Quando a chave usada for do tipo K_a^{-1}, temos uma mensagem assinada (ver 2.3.5).

- Mensagens compostas por outras submensagens têm seus elementos separados por , (vírgula). São exemplos de mensagens compostas:

$$A, B, N_a$$
$$A, N_a, \{N_b\}K_{bs}$$
$$\{K_{ab}, A, N_b\}K_{bs}$$

- Uma especificação em notação padrão é uma seqüência finita de sentenças da forma

$$i.P \rightarrow Q : M$$

onde i é um número indicado a ordem na qual as sentenças devem ser lidas, P e Q são agentes quaisquer, M é uma mensagem (composta ou não) e o símbolo $\rightarrow$ representa a ação de envio de mensagens.

A descrição acima não é uma definição formal de uma linguagem, apenas uma convenção sintática para o entendimento de "especificações" em notação padrão.

O significado das sentenças em notação padrão está associado diretamente à simbologia que representa os principais objetos de um protocolo de criptografia (agentes, *nonces*, chaves etc.) e na indicação de que mensagens são trocadas a cada passo do protocolo. Essas indicações são feitas em tão alto nível que deixam margens a diferentes entendimentos do protocolo, o que pode vir a causar falhas durante o processo de implementação.

Carlsen (1994a), em sua taxonomia de falhas em protocolos de criptografia, classifica esse tipo geral de falha como **falhas dependentes de implementação**. Esse é o caso quando uma especificação de um protocolo, seja ela formal ou não, leva a diferentes implementações, com pelo menos uma delas contendo uma falha e pelo menos uma delas não contendo tal falha.

Para exemplificar o uso da notação padrão e suas deficiências, vamos fazer uma análise do protocolo de autenticação da ISO, como descrito em Carlsen (1994b, p. 105). O

protocolo é descrito em notação padrão como:

1. $A \rightarrow B : N_a$
2. $B \rightarrow A : \{N_a\}K_{ab}, \{N_b\}K_{ab}$
3. $A \rightarrow B : N_b$

Esse protocolo tem como objetivo a autenticação mútua entre o iniciador e o respondedor. Para que o protocolo funcione, é preciso que os agentes A e B compartilhem antecipadamente a chave K_{ab}. Como primeiro ponto fraco da notação padrão, podemos citar o fato de que essa suposição não está indicada, precisando ser acrescentada em linguagem natural.

Para entendermos melhor as limitações da notação padrão, vamos descrever o protocolo em linguagem natural. Observe os vários pontos que a especificação em notação padrão é incapaz de representar.

1. O agente A (iniciador) gera o *nonce* N_a e envia para o agente B (respondedor). Na realidade, a mensagem sendo enviada não consiste somente no *nonce* N_a. A identidade do agente iniciador deve também ser enviada, caso contrário, o respondedor não saberia para qual agente responder. A seguinte suposição poderia ser assumida:

   > Sempre que um agente A envia uma mensagem para o agente B, então a identidade de A e de B são enviadas juntamente com a mensagem indicada após o símbolo **:** (dois pontos).

   Porém, essa suposição não é usada de maneira consistente. Em alguns protocolos, como no caso acima, ela parece valer, já em outros, como como no caso do protocolo Needham-Schroeder (NEEDHAM; SCHROEDER, 1978), as mensagens enviadas contêm explicitamente a identidade do agente emissor. Somente observando a descrição em notação padrão, não conseguimos perceber se o projetista do protocolo se valeu da suposição acima ou está cometendo um erro de sub-especificação.

2. O respondedor criptografa o *nonce* recebido com a chave compartilhada com o agente cuja identidade vem juntamente com a mensagem (*nonce*). É neste ponto que a afirmação de que a identidade do transmissor deve fazer parte da mensagem enviada justifica-se. Se tal afirmação não fosse verdadeira, então o respondedor não poderia identificar qual chave utilizar na criptografia, e ainda não saberia para qual agente enviar a resposta. Nesse mesmo passo, o respondedor deve também criar um

nonce, criptografá-lo com a mesma chave citada anteriormente e, finalmente, enviar as duas mensagens criptografadas de volta para o agente cuja identidade veio junto do *nonce* recebido.

3. Neste passo, o agente iniciador deve descriptografar a primeira das submensagens da mensagem recebida (que possui duas submensagens) usando a chave compartilhada com o agente para o qual a primeira mensagem (do passo 1) foi enviada. Depois de descriptografada, essa mensagem deve ser comparada àquela enviada no primeiro passo. Caso os valores sejam os mesmos, o iniciador tem a garantia de que essa parte da resposta pertence a essa execução do protocolo. Continuando, o iniciador deve também descriptografar a segunda submensagem usando a mesma chave compartilhada anteriormente e enviar o resultado de volta para o mesmo agente para o qual a mensagem do passo 1 foi enviada. Percebe-se a incapacidade da notação padrão em representar as ações de descriptografar e comparar mensagens.

4. Este último passo sequer é mencionado na notação padrão. Nele, o agente respondedor deve conferir a mensagem recebida com o *nonce* que foi criado no passo 2. Note que, para isso, esse agente deveria ter armazenado tal valor até o momento da verificação. De forma análoga, o iniciador deveria ter armazenado o valor do *nonce* enviado no passo 1 para posterior verificação no passo 3. A ação de armazenamento de *nonces* também pode ser entendida como implícita na notação padrão. Neste caso, parece não haver um uso inconsistente dessa suposição. Em protocolos onde um agente gera mais de um *nonce*, talvez a menção explícita de qual *nonce* está sendo armazenado e verificado seja necessária.

Continuando com a análise, vamos conhecer uma falha existente no protocolo de autenticação da ISO.

Para que tal falha aconteça, é preciso que o ambiente no qual o protocolo esteja envolvido permita que um mesmo agente possa desempenhar o papel de iniciador e respondedor, inclusive simultaneamente. Quando uma falha decorre desse tipo de suposição, Carlsen (1994a) a classifica como uma **falha de oráculo com múltiplos papéis**. Vamos descrever o cenário de ataque, ou seja, a seqüência de troca de mensagens em que tal falha ocorre (CARLSEN, 1994b, p. 105). Para isso, usaremos também a notação padrão com algumas novas convenções. A personificação de um agente será indicada com o símbolo do agente personificado (na verdade, a letra minúscula correspondente) sendo subscrito ao símbolo do agente personificador, assim $\boldsymbol{C_b}$ representa o agente C se fazendo passar por B, ou seja, C usando, como endereço de origem das mensagens enviadas por ele, a

identidade do agente B. Para representar os passos de diferentes execuções, vamos preceder o número do passo com o número da execução. Sendo assim, 2.1 indica o passo 1 da execução 2. Para diferenciar os *nonces* gerados pelo agente A nas duas execuções, usaremos um apóstofro.

1.1. $A \rightarrow C_b : N_a$
2.1. $C_a \rightarrow A : N_a$
2.2. $A \rightarrow C_b : \{N_a\}K_{ab}, \{N'_a\}K_{ab}$
1.2. $C_b \rightarrow A : \{N_a\}K_{ab}, \{N'_a\}K_{ab}$
1.3. $A \rightarrow C_b : N'_a$
2.3. $C_b \rightarrow A : N'_a$

O que acontece é que o agente maligno C aproveita-se do formato das mensagens trocadas no protocolo e do comportamento mecânico do agente respondedor. Tal agente, ao receber uma mensagem, simplesmente usa a chave compartilhada K_{ab} para criptografá-la, sem verificar a possibilidade dele mesmo ter sido o criador de tal mensagem e estar atuando também como iniciador em outra execução do protocolo.

Neste ponto, vamos discutir outra limitação da notação padrão. Observando a especificação inicial do protocolo, não fica claro que os símbolos A e B estão na verdade representando quaisquer dois agentes, e não agentes particulares. A especificação em notação padrão permite uma interpretação errônea de que, uma vez que uma mensagem seja enviada para um agente, somente esse tenha a possibilidade de recebê-la. O ataque descrito acima faz uso da suposição de que A e B representam um iniciador e um respondedor quaisquer, e que, uma vez que uma mensagem seja enviada, é razoável assumir a possibilidade de qualquer outro agente recebê-la.

Note que a falha descrita acima não é proveniente de nenhuma das limitações da capacidade de representação da notação padrão, ou seja, não se trata de uma falha por sub-especificação, ou como previamente discutido, de uma falha dependente de implementação. A causa da falha do protocolo é sua própria estrutura, *i.e.*, o formato das mensagens trocadas permite uma manipulação que pode subverter o protocolo. Nosso objetivo foi demonstrar a inabilidade da especificação padrão em descrever, mesmo que informalmente, algumas ações do protocolo. Na próxima seção, estudaremos um caso onde a sub-especificação do protocolo é a real causadora de uma falha dependente de implementação.

Para corrigir a falha no protocolo de autenticação da ISO apresentada acima, propomos a seguinte especificação em notação padrão.

$$
\begin{aligned}
&1.\ A \rightarrow B : \{A, N_a\}K_{ab} \\
&2.\ B \rightarrow A : N_a, \{N_b\}K_{ab} \\
&3.\ A \rightarrow B : N_b
\end{aligned}
$$

Agora, o respondedor tem como certificar-se de que a mensagem recebida no passo 2 não foi criada por ele mesmo, bastando para isso que ele descriptografe a mensagem $\{A, N_a\}K_{ab}$ e compare a identidade A com a sua própria. Essa é mais uma ação que não está especificada na notação padrão.

2.6 Falhas em Protocolos de Autenticação e Distribuição de Chaves

Nesta seção, vamos analisar alguns exemplos clássicos de protocolos da classe de autenticação e distribuição de chaves. Nosso objetivo principal é familiarizar o leitor com o tipo de raciocínio que deve ser efetuado na análise desse tipo de protocolo. Além de conhecer os protocolos, vamos descrever suas falhas em linguagem natural. Tais descrições poderão ser futuramente comparadas às descrições formalizadas do capítulo 5. Outras fraquezas da notação padrão como linguagem de especificação também serão evidenciadas.

Um protocolo bem conhecido, o qual se propõe a realizar a distribuição de uma chave de sessão com autenticidade, é o Otway-Rees, descrito em Boyd (1990). O protocolo envolve os agentes A e B e um agente servidor S, cuja função é criar e intermediar a distribuição da chave de sessão. Esse protocolo possui uma falha que pode ser classificada de acordo com Carlsen (1994a) como uma **falha de ação interna**. Vamos descrever o protocolo e tal falha. Inicialmente, S compartilha as chaves K_{as} e K_{bs} com os agentes A e B respectivamente. Em notação padrão, o protocolo Otway-Rees pode ser escrito como:

$$
\begin{aligned}
&1.\ A \rightarrow B : M, A, B, \{N_a, M, A, B\}K_{as} \\
&2.\ B \rightarrow S : M, A, B, \{N_a, M, A, B\}K_{as}, \{N_b, M, A, B\}K_{bs} \\
&3.\ S \rightarrow B : M, \{N_a, K_{ab}\}K_{as}, \{N_b, K_{ab}\}K_{bs} \\
&4.\ B \rightarrow A : M, \{N_a, K_{ab}\}K_{as}
\end{aligned}
$$

Consideremos o motivo de enviar as informações não criptografadas M, A e B no passo 2. Os projetistas do protocolo consideram M como um **identificador de sessão**, o qual é

usado por A e B para identificar se a mensagem recebida de S refere-se à execução corrente do protocolo. A razão para enviar as identidades de A e B nos passos 1 e 2 é para que S seja capaz de saber com que chaves descriptografar as submensagens criptografadas no passo 2. Uma parte essencial do protocolo é que S cheque se essas submensagens estão corretas. Em Burrows, Abadi e Needham (1990), é dito que S deve verificar se os componentes M, A e B são os mesmos nas duas submensagens criptografadas:

$$\{N_a, M, A, B\}K_{as} \qquad \text{e} \qquad \{N_b, M, A, B\}K_{bs}$$

Os mesmos autores acreditam que uma afirmação similar é feita pelos próprios projetistas do protocolo. Sendo assim, o seguinte entendimento, a respeito das ações que devem ser executadas por S no passo 3, é possível.

(a) S usa os identificadores A e B para escolher as chaves para descriptografar as submensagens criptografadas do passo 2.

(b) S checa se as submensagens M, A e B contidas nos blocos criptografados com K_{as} e K_{bs} possuem o mesmo valor.

(c) S cria a chave de sessão K_{ab} e a criptografa, juntamente com os respectivos *nonces*, usando as mesmas chaves descritas em (a).

Surpreendentemente, se S fizer somente essas verificações, então o protocolo pode ser considerado completamente inseguro. Qualquer agente C pode personificar qualquer outro agente que faça o papel de respondedor. Para isso, C deve simplesmente gerar seu próprio *nonce* e enviar a seguinte mensagem no segundo passo do protocolo:

$$C_b \rightarrow S : M, A, C, \{N_a, M, A, B\}K_{as}, \{N_c, M, A, B\}K_{cs}$$

Seguindo-se as ações descritas em (a), (b) e (c) acima, vê-se que a mensagem seria considerada correta por S, que daria continuidade a execução do protocolo, levando C a conhecer a chave de sessão K_{ab}, a qual foi concebida para comunicação entre A e B. Ao término de tal execução subvertida, o agente A irá pensar que compartilha a chave K_{ab} com B, o que não seria verdade. Uma vez que esse ataque tenha sido demonstrado, fica óbvio que S deve também comparar os valores de M, A e B das submensagens criptografadas com os valores daquelas enviadas em texto aberto. É interessante descobrir que esse protocolo depende criticamente de informações que são transmitidas em texto aberto, e não criptografadas.

A notação padrão procura descrever precisamente os dados enviados pelos agentes em cada passo do protocolo, mas sequer menciona as ações internas a serem executadas. Esse é um dos principais motivos pelos quais as especificações em notação padrão são sempre acompanhadas de explanações em linguagem natural. Nessas explanações, as principais ações internas, muitas vezes críticas para a funcionalidade do protocolo (ver 2.5), são apresentadas informalmente, ou simplesmente não são apresentadas, o que pode levar um implementador a cometer uma falha dependente de implementação.

Carlsen (1994c), sugere regras para deduzir as ações internas a partir da sintaxe da notação padrão. Tais regras podem ser descritas informalmente como:

- **Regra 1** – Para cada submensagem M enviada por um agente A num passo i, se o agente não recebeu a mensagem M anteriormente, então é assumido que ele criou e armazenou M antes do passo i.
- **Regra 2** – Se um agente receber a submensagem M pela primeira vez, ele deve checar seu tipo e armazenar seu valor.
- **Regra 3** – Se um agente subseqüentemente receber a submensagem M, ou seja, recebê-la mais de uma vez, então ele deve checar se o valor recebido é o mesmo que foi armazenado quando ele recebeu a submensagem M pela primeira vez.

O autor afirma que as regras são exaustivas, ou seja, cobrem todas a ações internas relevantes à segurança do protocolo. Essas regras são usadas pelo autor para transformar automaticamente uma especificação em notação padrão, numa especificação escrita numa extensão à linguagem CKT5, proposta em Carlsen (1993). O resultado desse estudo é que parece ser possível construir uma especificação formal e completa a partir de uma especificação numa linguagem semi-formal (notação padrão), baseando-se somente na sintaxe de tal linguagem e observando-se as características implicitamente presentes. Para contradizer essa conclusão, podemos citar o exemplo da falha no protocolo Otway-Rees, descrita anteriormente. Se observarmos a descrição do protocolo em notação padrão, poderemos perceber que, no passo

$$B \rightarrow S : M, A, B, \{N_a, M, A, B\}K_{as}, \{N_b, M, A, B\}K_{bs}$$

o agente S recebe as submensagens M, A e B pela primeira vez. Segundo a regra 2 acima, tudo que deve ser feito é checagem de tipo e o armazenamento dos valores. Porém, como foi demonstrado acima, a comparação dos valores das submensagens M, A e B que estão em texto aberto, com as submensagens M, A e B que estão criptografadas, é uma ação

interna crítica para a segurança do protocolo. Essa ação interna passaria desapercebida pelo processo de transformação de especificações proposto por Carlsen, resultando numa especificação em uma linguagem agora formal, porém ainda incompleta com respeito a descrição das ações internas relevantes à segurança do protocolo.

Burrows, Abadi e Needham (1990), ao comentar as qualidades de sua linguagem para análise de protocolos, apresentam a análise do protocolo Needham-Schroeder (NEEDHAM; SCHROEDER, 1978). Esse é mais um protocolo de autenticação e distribuição de chaves que tem como suposição inicial o fato de o iniciador (A) e de o respondedor (B) compartilharem inicialmente as chaves K_{as} e K_{bs}, respectivamente, com o servidor S, responsável pela geração e distribuição da chave de sessão K_{ab}. Esse protocolo contém uma falha que permite que um agente maligno leve dois outros agentes a acreditar que uma chave de sessão, a qual já foi utilizada anteriormente e encontra-se possivelmente comprometida, seja uma chave boa para comunicação. Esse tipo de falha é classificado por Carlsen (1994a) como **falha de frescor**. Vamos apresentar a especificação do protocolo em notação padrão, seguida de alguns comentários e de uma síntese da análise feita em Burrows, Abadi e Needham (1990).

1. $A \rightarrow S : A, B, N_a$
2. $S \rightarrow A : \{N_a, B, K_{ab}, \{K_{ab}, A\}K_{bs}\}K_{as}$
3. $A \rightarrow B : \{K_{ab}, A\}K_{bs}$
4. $B \rightarrow A : \{N_b\}K_{ab}$
5. $A \rightarrow B : \{F(N_b)\}K_{ab}$

Uma característica interessante desse protocolo é a de que somente o iniciador faz contato com o servidor de autenticação S, que por sua vez fornece para A uma chave de seção e um certificado criptografado com a chave compartilhada entre B e S. Esse certificado contém a chave de sessão K_{ab} e a identidade do agente iniciador. Após verificar que a chave recebida de S é recente, através da verificação do *nonce* N_a enviado no passo 1, e que tal chave serve para comunicação com B, o agente A envia o certificado para B. O agente B descriptografa o certificado e tem acesso a chave de sessão K_{ab}. Para garantir a presença de A na execução em trânsito, o agente B gera e envia o *nonce* N_b criptografado com a chave de sessão K_{ab}. A submensagem $F(N_b)$, presente no último passo do protocolo, representa o resultado de uma função de modificação qualquer, previamente estabelecida,

sobre o *nonce* N_b. O propósito da aplicação dessa função é diferenciar a mensagem enviada no passo 4 daquela enviada no passo 5.

Aplicando as regras de inferência da lógica proposta pelos autores, amplamente conhecida como **lógica BAN**, poderemos fazer deduções a respeito do conjunto de crenças das entidades que participam do protocolo. Os detalhes sobre as regras e como elas são aplicadas podem ser encontrados no trabalho original (BURROWS; ABADI; NEEDHAM, 1990). Vamos descrever informalmente as principais deduções da análise.

- Ao receber a mensagem $\{N_a, B, K_{ab}, \{K_{ab}, A\}K_{bs}\}K_{as}$ no segundo passo do protocolo, o agente A acredita que a submensagem K_{ab} foi criada por S, já que K_{as} é uma chave compartilhada somente por A e S, assumindo que que A é capaz de reconhecer as mensagens criadas por ele próprio. O agente A acredita ainda que K_{ab} é recente, uma vez que essa submensagem veio criptografada juntamente com o nonce N_a, enviado no primeiro passo.

- A regra da jurisdição da lógica BAN afirma que, se um agente A acredita que outro agente B acredita numa mensagem M, e ainda, A acredita que S tem jurisdição sobre M, então A deve também acreditar que M é verdade. O significado de acreditar numa mensagem e de uma mensagem ser verdade não parecem intuitivos e são pontos criticados em outros trabalhos, como em Nesset (1990). Usando-se o raciocínio da regra da jurisdição e a dedução do item anterior, chega-se a conclusão de que A acredita que a chave de sessão K_{ab} é boa para comunicação com o agente B. Isso acontece pois A acredita que a chave de sessão K_{ab} foi gerada por quem realmente deveria ser (o servidor S) e também que K_{ab} é recente.

- Prosseguindo com a análise, deduz-se que B acredita que a mensagem $\{K_{ab}, A\}K_{bs}$, enviada no passo 3, foi originalmente criada por S, já que vale a suposição inicial que K_{bs} é uma chave compartilhada somente por B e S.

A análise do conjunto de crenças de B para aí. Isso acontece pois não se consegue deduzir o resultado intermediário de que B acredita que a chave K_{ab} é recente, já que não há nenhum *nonce* no certificado que possa ser usado por B para garantir o frescor da chave de sessão. Sem esse resultado intermediário, não se consegue a dedução de que B acredita que K_{ab} é uma boa chave para comunicação com A. Do ponto de vista puramente formal, poderíamos ter inserido a suposição inicial de que o agente B acredita que a chave K_{ab} é recente e assim conseguir a dedução final desejada. Tal suposição, no entanto, é muito forte para a maioria das situações práticas, o que inviabiliza a utilização desse protocolo

da forma como apresentado. É interessante notar como a dedução da segurança desse protocolo está intimamente ligada às suas suposições iniciais, ou seja, segundo a lógica BAN, o protocolo poderia ser considerado correto se a suposição inicial de que B acredita que a chave K_{ab} é fresca fosse assumida. Podemos generalizar esse raciocínio também para o caso dos objetivos finais dos protocolos. Ou seja, a dedução de que um protocolo é seguro depende dos conjunto de suposições iniciais assumidas e de seus objetivos finais. Como parte dessa argumentação, citamos a seguinte afirmação:

> A notação padrão não descreve os objetivos pretendidos por um protocolo. Tal como as suposições iniciais, uma descrição precisa dos objetivos dos protocolos ajuda os usuários em potencial a escolher um protocolo apropriado aos seus requisitos, e ressaltam o que exatamente um protocolo pode alcançar. Em segundo lugar, análise formal é usada para provar que uma especificação de um protocolo, juntamente com um conjunto de suposições iniciais, obtêm seus objetivos, e uma descrição formal é requerida para fazer isso. (CARLSEN, 1994c, p. 3, tradução nossa).

2.7 Conclusões

Existem protocolos de criptografia para resolver diversos tipos de problemas relacionados aos requisitos de segurança apresentados na seção 2.2, desde a realização de eleições eletrônicas até problemas envolvendo dinheiro digital (SCHNEIER, 1996). Um problema particularmente interessante na área de comunicação de dados é o de autenticação e distribuição de chaves. Sabemos que os algoritmos de criptografia simétricos podem ser usados para obtermos confidencialidade e integridade, porém, para que esse modelo seja funcional, é preciso que os agentes envolvidos, e somente esses, compartilhem uma chave secreta. Como distribuir essa chave com autenticidade, confidencialidade e integridade é uma questão central em segurança na comunicação de dados. Como afirmado anteriormente, o problema tem inclusive um caráter circular, já que, para obtermos confidencialidade, autenticidade e integridade, precisamos distribuir uma informação (a chave secreta) através de um canal inseguro, mas com confidencialidade, integridade e autenticidade. Apesar da funcionalidade dos modelos de criptografia (ver 2.3), os mais diversos tipos de falhas podem surgir quando esses modelos são empregados na tentativa de prover os agentes de um canal seguro (CARLSEN, 1994a). Muitas dessas falhas são decorrentes da inabilidade dos métodos usuais de especificação, em particular da notação padrão, em descrever pontos críticos dos protocolos, conforme foi demonstrado nas seções 2.5 e 2.6.

A descoberta da complexidade e variedade de formas de ataque que vinham sendo descobertos alertaram a comunidade científica para a necessidade da aplicação de métodos formais para análise dos requisitos de segurança em protocolos de criptografia. No capítulo seguinte, vamos conhecer as principais abordagens propostas com esse propósito.

3 MÉTODOS FORMAIS PARA ESPECIFICAÇÃO E ANÁLISE DE PROTOCOLOS DE CRIPTOGRAFIA

Neste capítulo, apresentamos uma revisão bibliográfica sobre os métodos formais para análise de protocolos de criptografia e assuntos relacionados. Procuramos apresentar as principais características de cada trabalho, seu escopo e suas contribuições. Esse mesmo tipo de coletânea é feito por Clark e Jacob (1996).

3.1 Introdução

Em 1978, Needham e Schroeder propuseram um protocolo para resolver o problema da autenticação e distribuição de chaves em redes de computadores. Tal protocolo é descrito e comentado na seção 2.6. Na época, esse protocolo foi considerado correto, até que, em 1981, Denning e Sacco demonstraram que havia uma falha, classificada por Carlsen (1994a) como uma **falha de frescor**. Tal falha está relacionada à possibilidade de que, se um ataque de criptoanálise objetivando descobrir qual chave foi utilizada numa sessão já encerrada entre dois outros agentes for bem sucedido, então um agente maligno poderia personificar o iniciador em futuras sessões simplesmente enviando as mesmas mensagens da sessão anterior (ver 2.6).

O crescimento na variedade e complexidade das falhas, como mostrado na seção 2.6, e por Carlsen (1994b, cap. 2), gerou uma demanda pelo uso de métodos, formais ou não, que possam garantir a segurança dos protocolos.

Mas o que significa dizer que um protocolo é seguro? Como argumentado na seção 4.2, a resposta a essa pergunta está geralmente atrelada ao próprio protocolo (seus objetivos e suposições) e ao método de análise utilizado. Carlsen (1994b, p. 31) levanta ainda

outros questionamentos a respeito da análise de protocolos. Uma questão diz respeito à **completude** do método: será que ele captura todas as falhas que podem ser concebidas? Outra questão é a **praticabilidade** do método: os usuários têm garantias de que uma resposta sobre a segurança do protocolo será encontrada num tempo viável? A resposta para ambas as perguntas é não. Primeiramente, não é realista a suposição de que um método pode detectar todos os tipos de falhas, pois isso requer que o projetista do método tenha conhecimento sobre todas as formas com que um protocolo pode ser atacado. O melhor que podemos esperar é ter um método que detecte todas as falhas de acordo com uma taxonomia, como a proposta por Carlsen (1994a). Em segundo lugar, mesmo sendo os protocolos de criptografia geralmente constituídos de poucos passos, o número de diferentes formas com que esses passos podem ser implementados e, durante a operação do protocolo, combinados de formas legais e ilegais (*i.e.*, ilegais do ponto de vista do projetista, mas legais quando consideradas as capacidades dos agentes malignos), torna praticamente impossível prover uma resposta absoluta à pergunta se o protocolo é ou não seguro. Carlsen (1994b, p. 32), apresenta ainda outras variáveis, além da completude e praticabilidade, as quais diferenciam os vários métodos de análise.

Apesar desse quadro pessimista, várias abordagens tem sido propostas para especificar e analisar protocolos de criptografia. Embora nenhuma delas ofereça garantias quanto a completude ou praticabilidade, muitas delas têm sido usadas com sucesso na descoberta de novas falhas e têm aumentado a confiabilidade em determinados protocolos.

Carlsen (1994b, p. 32) afirma ainda que, dada a diversidade dos métodos de análise, torna-se impraticável prover uma classificação exata de todos os métodos existentes. Apesar dessa argumentação, o autor cita diversas tentativas de classificação que dão ênfase a diferentes características dos métodos. O próprio autor também utiliza-se basicamente de uma dessas classificações para apresentar seu resumo e comparação entre métodos (CARLSEN, 1994b, cap. 3).

Na seção seguinte, vamos citar alguns dos principais métodos de análise, seus escopos de atuação e seus principais resultados. Referências a outras publicações de interesse correlato também serão comentadas.

3.2 Classificação dos Métodos

Em Meadows (1992), a classificação clássica dos métodos de especificação e análise é apresentada. Naquele trabalho, a autora divide as abordagens em 4 grandes grupos:

1. Uso de linguagens de especificação e ferramentas de verificação as quais não foram desenvolvidas especialmente para modelagem e verificação de protocolos de criptografia;

2. Uso de sistemas especialistas, nos quais os projetistas podem desenvolver protocolos e investigar diferentes cenários;

3. Modelagem dos requisitos de famílias de protocolos através do emprego de lógicas para análise de crença e conhecimento;

4. Métodos formais baseados nas propriedades de reescrita de termos algébricos em que os protocolos podem ser modelados.

Em Meadows (1995), as principais características de cada uma dessas abordagens são apresentadas. Outras comparações entre métodos seguindo essa classificação podem ser encontradas em Carlsen (1994b), Rubin e Honeyman (1993), Gritzalis, Spinellis e Georgiadis (1999) e Clark e Jacob (1996).

3.2.1 Linguagens e Ferramentas não Específicas

Em Boyd (1992), a linguagem LOTOS (*Language of Temporal Ordering Specification*) é usada na especificação de protocolos de autenticação. Dois exemplos são explorados. No primeiro, o protocolo Needham-Schroeder (NEEDHAM; SCHROEDER, 1978) é minuciosamente especificado. Tal como fazemos na seção 4.6, o autor argumenta que sua especificação descreve vários aspectos não presentes na especificação em notação padrão. O segundo exemplo consiste numa especificação em LOTOS do protocolo de autenticação *One-Way*, presente na recomendação X.509 da *International Telecommunication Union* (ITU), comentada em Stallings (1995), Schneier (1996) e Kaufman, Perlman e Speciner (1995). Nenhum procedimento para detecção de falhas a partir das especificações em LOTOS é apresentado.

Um modelo para descrever uma arquitetura de comunicação segura é apresentado em Boyd (1993). O modelo é escrito usando-se a linguagem formal Z. Tal modelo, ao contrário da maioria dos outros formalismos citados anteriormente, não tem a intenção de servir como ferramenta de especificação ou análise de nenhum protocolo particular. Em vez disso, seu objetivo é descrever arquiteturas de segurança em termos das propriedades fundamentais dos algoritmos de criptografia (ver 2.3). Desta forma, o modelo pode ser

útil na fase de projeto de um sistema de segurança, antes mesmo de qualquer protocolo ser considerado.

3.2.2 Sistemas Especialistas

Um provador de teoremas para análise de protocolos de criptografia, denominado por seus autores de CRYPTOLOG, é apresentado em Decker e Piessens (1997). Trata-se de uma tentativa de implementar o modelo de raciocínio da lógica BAN (BURROWS; ABADI; NEEDHAM, 1990). Ao contrário do que seria natural, a entrada do provador não é uma especificação formulada na linguagem da BAN, e sim uma descrição numa linguagem que é uma mistura dos paradigmas procedural e declarativo. Um mapeamento das sentenças da BAN para as sentenças de tal linguagem é mostrado. O protocolo Kerberos, amplamente discutido na literatura (SCHNEIER, 1996; OPPLIGER, 1996; KAUFMAN; PERLMAN; SPECINER, 1995; HUGHES, 1995; STALLINGS, 1995; CHESWICK; BELLOVIN, 1994; CHAPMAN; ZWICKY, 1995) é analisado como exemplo.

De acordo com o afirmado em Rubin e Honeyman (1993), o sistema especialista para análise de protocolos de criptografia conhecido como **INTERROGATOR** (MILLEN; CLARK; FREEDMAN, 1987) recebe como entrada uma especificação de um protocolo e uma informação alvo. A saída é uma mensagem mostrando como um agente maligno pode obter a informação alvo. Para o INTERROGATOR, uma especificação de um protocolo é um conjunto de processos comunicantes, um para cada agente envolvido. Cada processo possui um conjunto de possíveis estados, e a transmissão de uma mensagem pode causar uma transição de estado no processo do agente correspondente. Cada processo mantém seu próprio estado atual. O sistema é baseado na abordagem de máquinas de estado finito (HOPCROFT; ULLMAN, 1979).

3.2.3 Lógicas Modais de Crença e Conhecimento

Dentre as diversas abordagens, a do uso de lógicas de crença e conhecimento é a que tem gerado um maior número de publicações e resultados positivos. O trabalho seminal dessa linha de pesquisa, batizado de **lógica BAN** pela comunidade científica (em menção às iniciais de seus autores) propõe uma lógica modal de crença que especifica e formaliza o raciocínio a respeito de propriedades funcionais de alto nível dos protocolos de criptografia (BURROWS; ABADI; NEEDHAM, 1990). Esse foi o primeiro sistema útil de prova em lógica modal a ser descrito e aplicado a protocolos de criptografia. O método baseia-

se em provar teoremas, os quais representam os objetivos de um protocolo. Contrária a vários outros métodos que a antecederam e a sucedem, a lógica BAN é fácil de aprender e usar, e muito de sua popularidade deve-se a esse fato. Cada objetivo de um protocolo é geralmente provado com poucas aplicações das regras de inferência propostas. A BAN demonstrou a importância de uma especificação precisa tanto para as suposições iniciais quanto para os objetivos dos protocolos. A BAN foi utilizada por seus autores para descobrir várias falhas em protocolos importantes. Contudo, as várias críticas sofridas pela BAN (ABADI; TUTTLE, 1991; BOYD, 1993; GLIGOR et al., 1991; NESSET, 1990) trouxeram à tona deficiências lógicas e semânticas que a invalidam como método formal.

Uma série de extensões à lógica BAN são comentadas em Carlsen (1994b, cap. 3).

Para tentar corrigir as deficiências do processo de idealização proposto em Burrows, Abadi e Needham (1990), e perpetuado por seus seguidores (MAO; BOYD, 1993; ABADI; TUTTLE, 1991; GONG; NEEDHAM; YAHALOM, 1990; HWANG; CHU; KU, 1993), a proposta de Mao (1995) formaliza a passagem das mensagens da notação padrão para formulação requerida pelos métodos baseados na lógica BAN. Trata-se de uma técnica baseada em regras para transformação sintática entre especificações. A idéia principal é refinar um passo maior de transformação de mensagens, existente nos métodos baseados na BAN, em passos menores e mais fáceis de serem entendidos. São demonstrados alguns exemplos desse processo.

Os autores Hwang, Chu e Ku (1993) propõem um sistema de verificação baseado em lógica que é uma extensão da lógica BAN. O propósito desse sistema é verificar ambos os aspectos de corretude dos protocolos: funcional e de segurança. O sistema funciona basicamente da seguinte forma. Uma especificação do sistema é criada a partir do processo de **idealização** (passagem da notação padrão para as sentenças da BAN). Nessa especificação, devem constar os objetivos do protocolo, suas suposições e as sentenças que representam o próprio protocolo. As novas regras de inferência, diferentes das originalmente propostas na BAN, são então aplicadas de forma recorrente em duas fases. Na primeira, tenta-se demonstrar que o requisito de confidencialidade das mensagens transmitidas é atendido, ou seja, que o protocolo especificado não contradiz as suposições iniciais. Na segunda fase, a tentativa é de provar que os objetivos do protocolo podem ser alcançados. Como críticas a esse trabalho podemos citar:

1. Nenhuma extensão à linguagem da lógica BAN é proposta, fazendo com que o sistema tenhas as mesmas limitações de escopo da lógica BAN;

2. Como amplamente discutido na literatura (ABADI; TUTTLE, 1991; BOYD, 1993; GLIGOR et al., 1991; NESSET, 1990), o processo de idealização não é intuitivo, e na formulação das novas sentenças, aspectos relevantes para a análise e segurança do protocolo podem deixar de ser representados.

Uma abordagem lógica diferente daquelas baseadas na BAN é apresentada em Woo e Lam (1993a). Diferentemente do trabalho original da BAN, a linguagem apresentada possui semântica bem definida num modelo de computação. Duas propriedades básicas para caracterizar a corretude de um protocolo são apresentadas: correspondência e segredo. Informalmente, **correspondência** significa que quando um agente termina com sucesso sua atuação no protocolo, então o agente autenticado deve ter estado presente na troca de mensagens que se antecedeu. **Segredo** é a propriedade que um protocolo pode possuir significando que determinadas informações não são acessíveis aos agentes malignos. São apresentados trechos de protocolos que exemplificam esses conceitos. Construções da linguagem proposta podem ser usadas para deduzir se um determinado protocolo possui as propriedades de correspondência e segredo. O protocolo Otway-Rees, discutido na seção 2.6, é usado como exemplo da aplicação do método. Esse trabalho é estendido em Woo e Lam (1994). O conjunto de regras de prova (na verdade axiomas escritos na linguagem proposta) é estendido para representar dois novos conceitos: **correspondência restrita** e **equivalência**. Um protocolo de autenticação baseado em criptografia assimétrica proposto pelos autores é usado como exemplo do processo de análise.

3.2.4 Reescrita de Termos Algébricos

O maior representante dessa linha é, sem dúvida, o analisador do *Naval Research Laboratory* (NRL), trabalho de constante evolução descrito em Meadows (1991, 1992) e Syverson e Meadows (1993). De acordo com o resumo de Carlsen (1994b, p. 37), para utilizar o NRL, cada sistema de criptografia é formulado com regras de reescrita de termos, onde, por exemplo, uma criptografia seguida de uma descriptografia com a mesma chave se cancelam. O protocolo deve ser então descrito como um conjunto de regras que especificam sob que condições novos termos podem ser gerados. Finalmente, os objetivos do protocolo devem ser sentenciados para descrever requisitos de confidencialidade e autenticidade. Diferentes linguagens são usadas para descrever os sistemas de criptografia, os protocolos e seus objetivos (SYVERSON; MEADOWS, 1993). Uma das características marcantes desse sistema é a capacidade de especificar ações internas, tais como testes que os agentes devem realizar nas mensagens que recebem. O NRL é uma ferramenta semi-automática, escrita

em PROLOG (CLOCKSIN; MELLISH, 1987), e que, uma vez carregada com as regras de reescrita de termos, com a especificação do protocolo e com seus objetivos, tenta achar uma seqüência de regras que caracterizam uma falha, ou seja, que vão de encontro aos objetivos especificados. Se tal tentativa não for bem sucedida, tem-se a indicação de que o protocolo especificado não pode ser subvertido pelas capacidades dos agentes malignos modeladas nas várias regras de reescrita de termos. Uma limitação citada pela própria autora em Meadows (1991) é a de que sistemas de reescrita de termos somente não são capazes de detectar falhas de frescor (CARLSEN, 1994a). Uma vez que a suposição de que os sistemas de criptografia são ideais, ou seja, ignorando a criptoanálise (ver 2.3), nenhuma regra pode ser usada para determinar em que ponto uma mensagem, a qual já era considerada como confidencial, acabou sendo conhecida por um agente maligno. Por outro lado, se existe a suspeita de que um protocolo possui uma falha de frescor, então um cenário pode ser construído onde um agente maligno é capacitado a conhecer uma mensagem confidencial, a qual ele normalmente não teria acesso. Uma análise do protocolo em questão nesse novo cenário vai revelar se aquela nova capacidade oferecida (propositadamente) ao agente maligno é suficiente para comprometer os objetivos do protocolo. Em Meadows (1991), essa técnica é utilizada para demonstrar uma falha de frescor num protocolo de compartilhamento de recursos. Uma técnica semelhante a essa é representada por um de nossos axiomas (TI), apresentados na seção 4.5. A principal diferença é que não precisamos oferecer uma nova capacidade aos agentes malignos, pois nosso axioma representa a possibilidade desses adquirirem uma informação confidencial (chave de sessão) uma vez que a execução em que tal informação foi transmitida já tenha sido encerrada.

3.2.5 Outros Métodos e Trabalhos Relacionados

Em Clark e Jacob (1996), além da já citada coletânea de referências bibliográficas com comentários, são apresentados conceitos básicos de criptografia, tipos de protocolos e falhas, uma revisão da classificação de Meadows (1995), e o mais interessante, uma biblioteca de protocolos agrupados por suas principais características.

Uma série de princípios para projeto de protocolos de criptografia são apresentados em Abadi e Needham (1996). Os autores não determinam quais são ou deveriam ser os objetivos dos protocolos, nem como eles devem ser especificados e verificados. Trata-se de explicação sobre quais técnicas podem ser usadas para combater determinados tipos de ataques. Os autores afirmam que seu conjunto de princípios não é necessário nem

suficiente para garantir a construção de protocolos seguros. Com certeza, tal afirmação é modesta, dada a variedade e riqueza dos casos cobertos nesse trabalho.

Em Mao e Boyd (1994), uma série de protocolos que empregam criptografia simétrica são analisados. Várias deficiências na especificação, projeto e análise de tais protocolos são apresentadas. Algumas dessas deficiências são consideradas responsáveis pela completa falta de segurança do protocolo, outras revelam apenas fraquezas que podem ser exploradas pelos agentes malignos. O problema básico identificado é a incompletude das especificações. Uma metodologia para desenvolvimento de protocolos seguros é proposta. Os autores citam ainda a possibilidade de extensão de seu trabalho para o tratamento de protocolos baseados em sistemas de criptografia assimétricos.

Os autores Woo e Lam (1994) descrevem a lição aprendida através da descoberta de uma falha num protocolo proposto por eles mesmos numa publicação anterior (WOO; LAM, 1992). O protocolo e sua falhas são descritos. A falha é descrita como sendo introduzida pelo processo de derivação do protocolo a partir da chamada **versão de informação completa**, dada como correta. Um protocolo de autenticação é dito estar sob a forma de informação completa se o iniciador e o respondedor incluem em todas as mensagens emitidas todas as informações que cada um deles acumulou até o aquele momento através da troca de mensagens. O **princípio da informação completa** é introduzido e exemplos são usados para demonstrar como tal princípio, quando seguido, pode levar a concepção de protocolos seguros. Heurísticas são propostas para simplificar protocolos que estejam sob a forma de informação completa.

Uma visão do ciclo completo de desenvolvimento de um protocolo de autenticação é estudada em Woo e Lam (1993b). O ciclo é dividido em 4 fases: projeto, especificação, verificação e implementação. Na fase de projeto, o protocolo sofre um processo de refinamento sucessivo, exemplificado pelos autores. A especificação e análise são realizadas pelo método proposto em Woo e Lam (1993a) e estendido em Woo e Lam (1994). Para a fase de implementação, os autores propõe o uso da GSS-API, uma interface de programação para implementação de protocolos de segurança bastante utilizada em outras tecnologias (OPPLIGER, 1996). Um mapeamento da especificação do protocolo para a GSS-API é mostrado. Como os próprios autores afirmam, a perspectiva global apresentada nesse trabalho contribui para a diminuição da distância entre entre a teoria e a prática no que se refere ao projeto e análise de protocolos de criptografia.

4 FORMALIZANDO ATAQUES

Neste capítulo, apresentamos uma teoria de primeira ordem para formalizar a descrição de falhas previamente conhecida ou ainda testar a presença de uma possível falha. Em ambos os casos, especificações do protocolo e da falha, ou possível falha, devem ser previamente construídas. Quando aliada a um sistema de prova automática de teoremas, tal teoria pode ser usada para testar a presença de um grande número de possíveis falhas. Seja a análise manual ou automática, quando uma falha é encontrada, um cenário de ataque (execução com falha) é formalmente apresentado. Para uma introdução à lógica de primeira ordem, sugerimos Kelly (1997), Fitting (1990), Ebbinghaus, Flum e Thomas (1984), Manna (1974) e Casanova, Giorno e Furtado (1987).

Iniciamos com a estrutura geral do procedimento de análise. Em seguida, discutimos questões divergentes e de suma importância a respeito de protocolos de criptografia e seus métodos de especificação e análise. Nossas opiniões sobre tais questões serviram como princípios para a construção da teoria. Logo após, segue-se a apresentação da sintaxe e semântica da linguagem de primeira ordem usada para especificar protocolos e falhas. Conhecimento tácito a respeito do ambiente hostil de comunicação que envolve os protocolos de criptografia é modelado como um conjunto de axiomas. Por fim, comentários sobre porque a linguagem proposta não sofre das mesmas limitações da notação padrão (CARLSEN, 1994c) e exemplos sobre construção de especificações e representações de falhas usando tal linguagem são apresentados.

4.1 Estrutura Geral

A maioria das abordagens lógicas para análise de protocolos são baseadas em prova de teoremas. Geralmente, tenta-se formalmente deduzir a corretude de um protocolo através de provas de propriedades desejáveis (BURROWS; ABADI; NEEDHAM, 1990; CARLSEN, 1994c; ABADI; TUTTLE, 1991; MAO; BOYD, 1993; GONG; NEEDHAM; YAHALOM, 1990), ou mostrar uma falha deduzindo-se propriedades não desejáveis (ABADI; TUTTLE,

1991; MAO; BOYD, 1993), ou ainda demonstrar que o protocolo não atinge um determinado conjunto de objetivos (CARLSEN, 1994c). A abordagem deste trabalho também é baseada em prova de teoremas. Contudo, não estamos interessados em deduzir propriedades (boas ou más) a respeito dos protocolos. Nossa proposta é a detecção de **cenários de ataque**, nos quais fiquem evidenciados formalmente todos os passos realizados pelos agentes e que caracterizem falhas. Fórmulas vão representar ações realizadas pelos agentes, alterações no estado de conhecimento de agentes, relações entre mensagens ou ainda relações de tempo entre passos de uma mesma execução de um protocolo e entre execuções distintas. Estaremos utilizando o termo **abordagem** preferencialmente a **método** uma vez que não há nada de novo em utilizar lógica de primeira ordem como modelo formal para representação de conhecimento e análise e especificação formal, apenas estaremos explorando sua utilização no campo dos protocolos de criptogafia. O esquema geral de detecção de falhas é o descrito abaixo.

Sejam os seguintes conjuntos de fórmulas:

Δ Conjunto de axiomas que modela um ambiente hostil de comunicação. Essas fórmulas representam o conhecimento notório sobre a fragilidade dos canais de comunicação, habilidade dos agentes malignos em interceptar e manipular as mensagens, propriedades dos sistemas criptográficos, capacidade dos agentes envolvidos, dentre outras características.

Φ Representa a especificação formal de um protocolo. As fórmulas desse conjunto são propostas pelo projetista do protocolo. Esse deve ter o cuidado de verificar se sua especificação exprime realmente arquivo que ele tem em mente, sob pena de uma falha ser (ou não) detectada, mesmo que ele estivesse precavido sobre essa possibilidade. O conjunto Φ pode ainda conter a especificação de mais de um protocolo, representando assim um ambiente multiprotocolar, onde mensagens relativas a execução de um dos protocolos podem acabar sendo tratadas como parte da execução de outro. Esta situação pode vir a demonstrar uma falha não detectada no caso da análise individual de cada protocolo.

Estamos interessados em demonstrar que determinadas fórmulas, que aqui representaremos por φ, as quais possuem determinadas restrições sintáticas (ver 4.7) e representam condições de falha, são deduzíveis a partir de $\Delta \cup \Phi$, ou seja:

$$\Delta \cup \Phi \vdash \varphi$$

As fórmulas φ devem ser estabelecidas pelo analista. É sua função determinar quando uma fórmula representa uma falha, de acordo com os objetivos do protocolo em questão.

Escolhemos como método de prova de teoremas o **cálculo de seqüentes** (SZABO, 1969; FITTING, 1990; EBBINGHAUS; FLUM; THOMAS, 1984). Nossa escolha foi baseada na facilidade com que esse método pode ser usado de forma independente de implementação e, principalmente, por termos proposto uma transformação sintática para as árvores de prova do cálculo de seqüentes que torna as demonstrações de falhas mais intuitivas, correspondendo a uma execução do protocolo (ver 5.2). Outros métodos, como a dedução natural (SZABO, 1969), resolução linear com função de seleção para cláusulas definidas, ou resolução LSD, e *Tableaux* (FITTING, 1990), poderiam também ser usados. A escolha do método de prova não é crítica, mas as propriedades de corretude e completude de nossa análise estão vinculadas à ela. Não estamos interessados somente no problema de decidir se as fórmulas φ são ou não teoremas de $\Delta \cup \Phi$. As seqüências que compõem a prova de φ, no caso dessa existir, vão representar exatamente o cenário de ataque que leva tal falha a acontecer. As fórmulas φ são escolhidas seguindo-se uma classificação mais genérica que a proposta por Carlsen (1994a). Nela, são considerados apenas os requisitos de segurança básicos, que possivelmente não estão sendo atendidos pelo protocolo em questão, a saber: integridade, confidencialidade, autenticidade e não-repudiação.

O cálculo de seqüentes produz provas mínimas, ou seja, nenhuma subprova desnecessária para a prova de um teorema irá fazer parte da árvore de prova de seqüentes. Dada a dificuldade de apresentação de árvores de provas do cálculo de seqüentes, dificuldade essa devida ao crescimento exponencial do tamanho das árvores, propomos no capítulo 5 uma maneira de apresentar árvores de prova de forma linear, ou seja, como uma seqüência finita de linhas. Chamamos esse mapeamento de uma árvore para uma série de linhas de **aplanamento** da árvore. Esse mapeamento não tem a intenção de ser um método de prova de teoremas, tratando-se apenas de uma modificação sintática para viabilizar a apresentação das provas num espaço restrito. Portanto, não encorajamos o desenvolvimento de provas de teoremas diretamente sob essa forma.

Dentro da classificação clássica proposta por Meadows (1992), a abordagem deste trabalho pode ser vista como uma mistura entre lógica e busca exaustiva. Das lógicas, herdamos a linguagem e todo um conjunto de mecanismos, corretos e completos, de prova de teoremas. Uma vez que o problema da satisfatibilidade de um conjunto de fórmulas, e portanto da prova de teoremas, é provado ser NP-completo (HOROWITZ; SAHNI, 1978; GAREY; JOHNSON, 1979), as técnicas de prova são normalmente implementadas usando-

se algoritmos de busca exaustiva, ou seja, no pior caso, um número exponencial de cenários são explorados até que um ataque seja detectado.

Para situar este trabalho num contexto maior, vamos responder aos questionamentos feitos por Carlsen (1994b, p. 30) ao diferenciar os vários métodos de análise:

- P. O método provê especificação somente ou especificação e análise?
  R. Especificação e análise. Tivemos ainda a preocupação de que as especificações sirvam como guia para os implementadores, evitando alguns erros provindos de especificações incompletas. Essa característica pode ser considerada um ganho, pois, do ponto de vista da análise somente, poderíamos ter deixado implícitos vários aspectos da especificação. Não o fizemos para forçar o autor dos protocolos a ser mais explícito, tornando a especificação mais rica, e, portanto, de maior utilidade para os implementadores. Além disso, a análise dos protocolos é simplificada pela explicitação de alguns aspectos na especificação, como veremos mais adiante.

- P. O método é informal ou formal?
  R. A linguagem que propomos tem sintaxe formal. Contudo, o modelo de computação associado à teoria é deveras simples, sendo constituído apenas de 3 conjuntos e suas relações (ver 4.4). A representação formal da realidade envolve, inevitavelmente, o informalismo, independentemente do escopo envolvido. O mapeamento do que é realidade (informal) para o que é modelo de computação (formal) é sempre descrito informalmente, não importando quantos modelos intermediários estejam envolvidos. Essa é uma característica de qualquer método formal. Ao invés de criar um modelo de computação mais elaborado (intermediário), processo esse que envolveria, inevitavelmente, descrições em linguagem natural, para depois fornecer uma semântica da linguagem sobre esse modelo, agora sim, através de um mapeamento formal, preferimos criar um modelo simples e definir uma semântica diretamente sobre ele. Despendemos esforços no sentido de representar os aspectos relevantes da realidade, sem deixar margens à interpretações dúbias.

- P. O método é baseado numa técnica de análise de propósito geral ou numa especialmente adaptada para análise de protocolos de criptografia?
  R. A classe das lógicas de primeira ordem tem o propósito geral de modelar conhecimento e o raciocínio sobre esse conhecimento. A abordagem proposta é, portanto, baseada numa técnica geral de especificação. Contudo, visando representar conhecimento sobre um assunto específico (no caso, protocolos de criptografia), construímos uma linguagem especialmente adaptada a esse propósito.

- P. O método é dirigido a uma busca exaustiva por falhas ou é baseado em prova de corretude?
  R. Diferentemente dos métodos baseados em lógica usuais, onde uma prova de corretude é buscada, o objetivo aqui proposto é demonstrar formalmente um cenário de ataque. Esse cenário é descrito pela prova de um teorema que descreve uma falha. Por tratar-se de um problema NP-completo, os métodos de prova de teoremas, e, em particular, o cálculo de seqüentes, são geralmente implementados através de busca exaustiva num espaço de soluções.

- P. Se o método é baseado em provas de corretude, ele é orientado por lógica ou álgebra?
  R. Abordagens orientadas por álgebra tendem a ser métodos baseados em provas de corretude em que uma busca exaustiva por falhas é realizada. O método aqui proposto herda características daqueles baseados em provas de corretude, uma vez que, para demonstrar cenários de ataque, precisamos provar teoremas. Essas provas são claramente orientadas pelos conceitos da lógica.

- P. Se o método é baseado em lógicas, qual o tipo de lógica utilizado? (modal, de predicados ou outro?).
  R. Apesar de fortes indicações que as lógicas modais são mais adequadas para representar crença e conhecimento, e portanto, para modelar uma prova de corretude do protocolo, usamos uma lógica de predicados clássica. A razão principal para isso é que nosso interesse está não em provar que um protocolo é correto, e sim na busca de cenários de ataque de demonstrem falhas.

- P. O método é suportado por ferramentas automáticas?
  R. O método de prova de teoremas aqui utilizado (cálculo de seqüentes) não é o mais adequado à implementação. Contudo, temos garantias de equivalência entre esse método e outros mais adequados à implementação, tais como resolução e *Tableaux*, isto pois ambos são provados serem corretos e completos (FITTING, 1990). Já que nosso método constitui-se basicamente na prova de teoremas, é plenamente passível de automatização.

4.2 Princípios de Projeto

Antes de apresentarmos formalmente a linguagem de primeira ordem utilizada que propomos, é necessário fazer algumas observações gerais sobre protocolos de criptografia

e seus métodos de especificação e análise mais conhecidos. Estas observações são resultados de nossas experiências e são interpretações que julgamos racionais a respeito de questões divergentes dentro da literatura sobre o assunto, ou simplesmente pontos sobre especificação e análise que desejamos ressaltar.

Especificações formais de protocolos de criptografia podem ser vistas basicamente sob dois aspectos:

- **Como parte do processo de análise formal** – Neste caso, o objetivo é formalizar provas de que o protocolo atente os requisitos básicos de segurança (ver 2.2). Dentre as várias propostas nessa linha (MEADOWS, 1992), as que usam linguagens lógicas têm tido os melhores resultados (CARLSEN, 1994c, p. 60). Contudo, parece haver ainda uma enorme lacuna entre especificações desse tipo e as reais implementações dos protocolos. Essa distância provoca uma desvalorização dos métodos de análise, já que, de nada adianta provar que um protocolo, ou melhor, que uma especificação funcional de alto nível atende os requisitos básicos de segurança a que se propõe, se o processo de implementação pode provocar o aparecimento de novas falhas.

- **Como parte do processo de desenvolvimento de software** – Aqui, a principal preocupação é livrar o implementador de interpretações errôneas sobre o comportamento do protocolo, o que poderia causar o que é chamado de **falhas de implementação** (CARLSEN, 1994c, p. 18). Nessa abordagem, uma vez que o protocolo tenha sido especificado, todas as interpretações que poderiam levar o implementador a cometer uma falha devem estar eliminadas.

Uns dos grandes desafios nesse campo está em prover os projetistas de protocolos com um método onde uma especificação formal servisse tanto para a análise de requisitos como para livrar o processo de desenvolvimento de possíveis **falhas dependentes de implementação**. Na abordagem proposta, esforços foram despendidos so sentido de nos aproximarmos do objetivo citado acima. Fornecemos meios para a formalização de alguns aspectos que tipicamente são os causadores de falhas de implementação, tais como a dependência de valores entre mensagens e entre ações (ver 2.6).

As lógicas para análise de protocolos de criptografia tipicamente não fazem menção explícita sobre tempo. Como exceções, podemos citar as lógicas CKT5, citada por Carlsen (1994b) e sua extensão proposta por Carlsen (1993). Na abordagem deste trabalho, as ações (envio de mensagens, conferência de *nonces* etc.) e alterações no estado de conhecimento dos agentes possuem como parâmetro o instante em que ocorrem dentro

de uma execução do protocolo. Vamos tratar esses instantes como variáveis discretas, que têm seu valor no conjunto dos naturais. Como estamos interessados em analisar múltiplas execuções, possivelmente simultâneas, usaremos também um identificador da execução, que também terá seu valor no conjunto dos naturais. O objetivo é poder avaliar se uma ação ocorre antes ou depois de outra ação dentro de uma mesma execução do protocolo. Consideraremos também uma relação de ordem entre as execuções, expressando explicitamente o caso onde uma execução inicia-se somente após o término de outra, bem como a possibilidade de uma execução paralela, já que pode acontecer de uma execução ser iniciada antes do término de outra. Essa diferenciação é importante, pois, nesse último caso, alguma informação que venha a ser adquirida por um agente maligno pode ser transportada para a execução paralela, servindo para completar a subversão de uma das execuções, o que não ocorre entre execuções que aconteceram uma após a outra, já que uma delas já estaria encerrada. Um exemplo de sentença que possui o tempo como parâmetro seria (em linguagem natural):

> No instante **3** da execução **4**, o agente $\boldsymbol{a}$ envia uma mensagem $\boldsymbol{m}$ para o agente $\boldsymbol{b}$.

Através do formalismo proposto, poderemos afirmar, por exemplo, que essa ação acontece antes de outra do tipo:

> No instante **4** da execução **4**, o agente $\boldsymbol{a}$ recebe do agente $\boldsymbol{b}$ a mensagem $\boldsymbol{m}$.

Assim como poderíamos garantir que tal ação aconteceu antes de uma terceira como:

> No instante **2** da execução **7**, o agente $\boldsymbol{a}$ armazena o *nonce* $\boldsymbol{N_a}$ (representação de um nonce).

Existe uma diferença conceitual importante entre os agentes e suas identidades. Um agente exerce ações dentro dos protocolos (envia e recebe mensagens, armazena *nonces* etc.), já sua identidade é um tipo particular de mensagem, que pode inclusive ser substituída por uma mensagem de outro tipo, como um *nonce*. Note que não faz sentido falar que um "agente" foi substituído por uma "chave". Assim como na maioria das lógicas, vamos usar as identidades dos agentes quando quisermos nos referenciar aos agentes em si. Uma sentença do tipo:

> O agente $\boldsymbol{a}$ envia para o agente $\boldsymbol{b}$ a mensagem $\boldsymbol{m}$.

deve ser entendida na realidade como:

> O agente cuja identidade é $\boldsymbol{a}$ envia para o agente cuja identidade é $\boldsymbol{b}$, a mensagem $\boldsymbol{m}$.

Essa abordagem pode parecer problemática, já que, podendo uma identidade ser substituída por outro tipo de mensagem, poderíamos obter sentenças que semanticamente não fazem sentido, do tipo:

> O agente $\boldsymbol{N_a}$, envia para o agente $\boldsymbol{K_{ab}}$ (representação de uma chave) a mensagem $\boldsymbol{m}$.

Porém, como será visto, a definição da sintaxe proposta foi elaborada de maneira que esse tipo de sentença não possa ser construída.

Consideraremos 3 tipos (*sorts*) de objetos: as identidades dos agentes, as mensagens e, juntos no último tipo, as execuções e seus instantes. As identidades dos agentes são um subtipo das mensagens. Objetos do tipo mensagem poderiam ainda ser subdivididos em outros tipos, como *nonces* e chaves. Contudo, não consideraremos essa subdivisão, pois queremos permitir que uma variável do tipo mensagem possa assumir um valor de qualquer um de seus subtipos, inclusive identidades de agentes. Por outro lado, variáveis do tipo identidade só devem assumir valores no domínio das identidades. Desta forma, esperamos formalizar o conceito de uma mensagem sendo substituída por outra, o que pode vir a demonstrar uma falha num protocolo.

Para tratar tal universo poli-sortido com uma linguagem de primeira ordem, Ebbinghaus, Flum e Thomas (1984) sugerem duas possibilidades. Na primeira, as estruturas têm vários domínios, cada um correspondendo a um tipo distinto, criando linguagens de primeira ordem poli-sortidas. Tais linguagens são construídas da mesma forma que as linguagens que tratam apenas um domínio, com a diferença de que nelas existem tantas classes de variáveis quantos forem os domínios da estrutura em questão e uma variável de uma classe só assume valores de seu domínio correspondente. Os símbolos funcionais e relacionais podem ser definidos sobre domínios específicos, não se aplicando a elementos de outros domínios. Na segunda possibilidade, as estruturas possuem apenas um domínio, onde estão inseridos todos os objetos de todos os tipos. São criados então tantos símbolos relacionais unários quantos forem os tipos existentes. Estas relações servem para classificar um elemento como de um determinado tipo. Os quantificadores de uma sentença podem então ser interpretados com relação a um determinado tipo, bastando para

isso que o símbolo relacional correspondente ao tipo em questão seja aplicado à variável quantificada.

Escolhemos a primeira abordagem por essa resultar numa linguagem sucinta e de semântica mais intuitiva, já que, por definição, as funções e relações ficam restritas a seus domínios de atuação. O mesmo não acontece na segunda abordagem, onde as sentenças tendem a aumentar de tamanho pela adição dos símbolos relacionais unários citados acima. Além disso, no segundo caso, as funções e relações devem ser estendidas a elementos não pertencentes aos seus domínios originais, o que, apesar de não exercer tanta influência na decisão (já que as sentenças podem ter seus quantificadores restritos a uma parte do domínio, através do uso das relações unárias citadas acima), resulta em uma semântica pouco intuitiva.

A diferença da abordagem usada neste trabalho em relação àquela proposta por Ebbinghaus, Flum e Thomas (1984) é que, em vez de criarmos uma linguagem poli-sortida da maneira usual, vamos apresentar uma sintaxe que restringe a formação de sentenças que não poderiam ser analisadas do ponto de vista semântico, já que, na sintaxe usual, nada nos impede de aplicar, por exemplo, símbolos relacionais de um domínio a elementos de outro domínio, resultando, neste caso, em sentenças sem significado plausível.

Tipicamente, protocolos são descritos apenas mostrando uma execução em particular, ou seja, as sentenças possuem instâncias de objetos (agentes, mensagens etc.), em vez de variáveis que podem assumir o valor de qualquer instância (do mesmo tipo). Não consideramos esse tipo de sentença adequado, pois com elas não conseguimos representar o fato de uma mensagem poder ser substituída por outra, ou de um agente personificar outro. Nas fórmulas que usaremos na especificação de protocolos, dificilmente existirão constantes. O uso de constantes impede a avaliação das sentenças do ponto de vista dos agentes malignos. Portanto, essas só devem aparecer quando o projetista quiser restringir o significado da sentença. Se convencionássemos que letras maiúsculas representam as variáveis e letras minúsculas as constantes, uma sentença do tipo:

> No primeiro passo do protocolo, o agente $\boldsymbol{a}$ envia para o agente $\boldsymbol{s}$ a mensagem $\boldsymbol{m}$.

indicaria que somente o agente $\boldsymbol{a}$ (constante) pode dar o passo inicial, o que pode não ser sempre o caso. Ao escrever

> No primeiro passo do protocolo, o agente $\boldsymbol{A}$ envia para o agente $\boldsymbol{s}$ a mensagem $\boldsymbol{M}$.

o projetista está expressando a possibilidade de qualquer agente poder enviar uma mensagem qualquer para o agente fixo $\boldsymbol{s}$, por isso o uso da constante $\boldsymbol{s}$. Outra fraqueza da notação padrão, como visto na seção 4.6, é a falta de poder para expressar a relação de dependência entre o conteúdo de uma mensagem enviada e o conteúdo de uma mensagem recebida. Além disso, a própria dependência entre as ações de envio e recebimento não fica bem especificada nessa notação. Acreditamos que com o uso de variáveis e de sentenças do tipo (o subscrito apenas diferencia as variáveis)

Envia M_1 se recebe M_1 concatenado com M_2

essas relações estarão melhor especificadas. O uso de uma mesma variável (no caso acima, M_1) em mais de um lugar numa mesma sentença, expressa a ligação (*binding*) entre os valores de tal variável. Esse é o comportamento real dos agentes, os quais muitas vezes, por não estarem aptos a dar um significado a uma mensagem recebida, devem tratá-la de forma mecânica, não levando em consideração seu significado ou conteúdo.

Em algumas lógicas para análise de protocolos, existe a preocupação em representar conhecimento de mensagens por parte dos agentes. Carlsen (1994b, p. 40) questiona o real significado dessa relação e propõe sua divisão em 3 conceitos mais simples:

- **Ciência** – Um agente pode (ou não) ter conhecimento a respeito da existência de uma mensagem, independentemente do fato de esse agente possuir a mensagem em si ou até mesmo de ele ser capaz de reconhecê-la, uma vez que venha a possuí-la.
- **Posse** – Um agente pode (ou não) possuir uma mensagem. Por possuir, entenda-se o conhecimento da real cadeia de bits que representa a mensagem. Tal conceito não está relacionado com o de ciência (apresentado acima), uma vez que um agente pode ter ciência de uma mensagem que ele não possui.
- **Interpretação** – Um agente pode (ou não) ter a habilidade de associar o conhecimento sobre a existência de uma mensagem (ciência) com sua posse. Em outras palavras, o fato de um agente possuir uma mensagem e saber (de acordo com a definição do protocolo) da existência de tal mensagem não garante que ele seja capaz de relacionar estes fatos e dar um significado à mensagem possuída. Se o agente não tem a habilidade de associar a posse com a ciência de uma mensagem, então ele não conseguirá usar a mensagem significativamente.

Essa divisão é importante nas abordagens puramente lógicas, onde o objetivo final é provar que um determinado protocolo possui determinadas propriedades ou que atende

um conjunto pré-estabelecido (pelo projetista) de objetivos. Neste mesmo trabalho, ao demonstrar as limitações da lógica BAN (BURROWS; ABADI; NEEDHAM, 1990), Carlsen (1994b, p. 47) cita um exemplo onde a não-diferenciação dos componentes do conhecimento torna impossível uma dedução da corretude de um protocolo. Como nosso interesse está na demonstração de cenários de ataques (e não em provas de corretude), não faremos distinção entre os componentes do conhecimento. Usaremos apenas um operador relacional, o qual terá semântica similar ao conceito de posse descrito acima. Posto de outra forma, as falhas que pretendemos demonstrar independem do significado dado às mensagens pelos agentes, com exceção de uma relação de compartilhamento de chaves, descrita a seguir na semântica da linguagem. Estamos particularmente interessados em representar o comportamento mecânico dos agentes, ou seja, a realização de ações mesmo sem o conhecimento completo das mensagens sendo manipuladas. Percebemos que aí está a origem da maior parte das falhas nos protocolos de criptografia. Dessa forma, estaremos considerando o pior caso, ou seja, a presença de qualquer indicio ou possibilidade de falha será tratado como uma falha.

Uma das maneiras de provar que um protocolo é "correto" consiste em, dada uma especificação que contenha um conjunto de objetivos do protocolo, mostrar que certas propriedades do protocolo atendem aqueles objetivos. O termo correto foi destacado pois não estamos nos referindo a uma característica absoluta, ou seja, o protocolo pode ser considerado correto num método de análise mas não em outro. Fica claro que o projetista é o responsável pela elaboração dos objetivos. Essa abordagem é a utilizada pela maioria das lógicas para análise de protocolos. Como não estamos interessados em uma prova de corretude do protocolo, preferimos não fornecer meios de representar seus objetivos. Cabe a quem estiver realizando a análise, verificar (*i.e.*, interpretar) se um teorema provado a partir da especificação, ou um cenário de ataque (uma prova de teorema), é contrário a algum dos objetivos do protocolo. Como será justificado na seção 4.7, cabe também ao analista a função de estabelecer quais sentenças representam situações de falha, baseado nos objetivos do protocolo. A abordagem sugerida irá fornecer o embasamento formal que permitirá a demonstração que tais sentenças são (ou não) conseqüência lógica da especificação proposta.

4.3 Alfabeto e Fórmulas

Iniciemos com algumas definições básicas.

Definição 4.1. *A um conjunto qualquer, não vazio, de símbolos chamamos* ***alfabeto****. Às*

*seqüências finitas de símbolos de um alfabeto $\mathcal{A}$ chamamos **strings** ou **cadeias** de $\mathcal{A}$. $\mathcal{A}^*$ denota o conjunto de todas as cadeias possíveis de $\mathcal{A}$.*

Como sugerem Ebbinghaus, Flum e Thomas (1984, p. 13), uma linguagem de primeira ordem é determinada pelo conjunto (possivelmente vazio) de símbolos funcionais, símbolos relacionais e constantes. Isto acontece pois os outros símbolos do alfabeto (conectivos, variáveis e constantes) podem ser considerados sempre presentes no alfabeto de qualquer linguagem, apresentando apenas algumas variações sintáticas (por exemplo, em algumas publicações, como em Fitting (1990) e Manna (1974), o símbolo $\supset$ é usado para denotar a implicação lógica, em outras, como é o caso de Ebbinghaus, Flum e Thomas (1984), o símbolo usado é $\longrightarrow$). As regras que definem os termos e fórmulas das linguagens também apresentam pequenas variações de sintaxe. A razão principal para as modificações que propomos a seguir na notação usual das lógicas clássicas é conseguir uma linguagem simples, intuitiva e que esteja de acordo com as observações feitas na seção 4.2. Para justificar melhor, vejamos um exemplo prático. Para formular uma sentença do tipo

> No instante 1, da execução 1, o agente $\boldsymbol{a}$ envia a mensagem $\boldsymbol{m}$, criptografada com a chave $\boldsymbol{k}$, para o agente $\boldsymbol{b}$.

na notação usual, poderíamos mapear um símbolo relacional $\mathbf{p}$ (de aridade 5) à relação de envio de uma mensagem, o símbolo funcional $\mathbf{c}$ (de aridade 2) à uma função de criptografia, e obter uma fórmula do tipo

$$\mathbf{p}(1, 1, a, b, \mathbf{c}(m, k))$$

Nossa proposta é usar símbolos relacionais e funcionais de sintaxe mais intuitiva (sempre que possível). Para o caso acima, teríamos $\longrightarrow$ como símbolo para a relação de envio de mensagem e o uso de $\mathbf{\{}$ e $\mathbf{\}}$ para representar a função de criptografia. Escreveríamos então

$$1.1\, a \rightarrow b : \{m\}k$$

que nos parece, além de mais intuitivo, mais próximo da notação usual dos protocolos de criptografia. Contudo, algumas vezes os símbolos usados podem não parecer tão intuitivos, como é o caso de $\boxtimes$, usado para indicar o armazenamento de mensagens (ver 4.4). Isso acontece pois nem sempre há um símbolo que nos remonte à relação ou função em questão.

Usar nomes maiores, do tipo $\boldsymbol{ARMAZENA}$ tornaria as fórmulas excessivamente longas. Para minimizar esse problema, escolhemos, sempre que possível, símbolos já usados na literatura, de preferência os de semântica semelhante à dos que propomos. Sendo assim, vamos definir uma linguagem desde o alfabeto. Estaremos, inclusive, estabelecendo as convenções sintáticas que serão usadas no decorrer do trabalho.

Definição 4.2. *O alfabeto $\mathcal{A}$ é o conjunto formado pela união dos seguintes subconjuntos de símbolos:*

$\mathcal{V}_1 = \{E_\circ\}$

$\mathcal{V}_2 = \{I_\circ\}$

$\mathcal{V}_3 = \{M_\circ, NC_\circ, CH_\circ\}$

$\mathcal{V}_4 = \{A, B, C, S, AG_\circ\}$

$\mathcal{S}_1 = \{,\text{(a própria vírgula)}, \Leftarrow, \Leftrightarrow,), (, :\}$

$\mathcal{S}_2 = \{F, N, K, K^{-1}, .\text{(ponto)}, \{,\}\text{(chaves)}, \langle, \rangle, [,]\}$

$\mathcal{S}_3 = \{\rightarrow, \leftarrow, <, \neq, \equiv, \boxtimes, \lhd, \leftrightarrow, \sim, ||\}$

$\mathcal{C}_1 = \{a, b, c, s\}$

$\mathcal{C}_2 = \{1, 2, 3, 4, \ldots\}$

$\mathcal{C}_3 = \{1^{||}, 2^{||}, 3^{||}, 4^{||}, \ldots\}$

$\mathcal{C}_4 = \{info, lixo\}$

Ou formalmente:

$$\mathcal{A} = \bigcup_{i=1}^{4} \mathcal{V}_i \cup \bigcup_{j=1}^{3} \mathcal{S}_j \cup \bigcup_{k=1}^{4} \mathcal{C}_k$$

O conjunto $\mathcal{V}_1$ compreende as variáveis do tipo execução. Usamos o $\circ$ para indicar que os símbolos desse conjunto podem, ou não, vir acompanhados de um subscrito para diferenciá-los. Vamos convencionar o uso de números inteiros naturais para esse propósito. Sendo assim, $\boldsymbol{E}$, $\boldsymbol{E_1}$ e $\boldsymbol{E_2}$ são exemplos de elementos de $\mathcal{V}_1$. Essa convenção também é válida para os demais casos onde o símbolo $\circ$ é encontrado subscrito.

$\mathcal{V}_2$ é o conjunto das variáveis que vão representar instantes de uma execução.

Em $\mathcal{V}_3$, estão as variáveis para mensagens. Usaremos $M_\circ$ para denotar mensagens em geral, $NC_\circ$ para denotar *nonces* e $CH_\circ$ para chaves. Porém, por se tratarem de variáveis de um mesmo tipo, ou seja, que podem assumir valores de um mesmo domínio, essa convenção não é obrigatória.

$\mathcal{V}_4$ contém as variáveis do tipo identidade de agentes. Esses símbolos, juntamente com os dos conjuntos anteriores, vão representar as variáveis de nossa linguagem.

Em $\mathcal{S}_1$, encontramos os conectivos lógicos e símbolos auxiliares. Dos conectivos usuais, utilizaremos apenas o **"*e*"** lógico, o qual denotaremos por **,** (vírgula), a implicação lógica reversa (a qual é normalmente denotada por $\longleftarrow$, mas que aqui escreveremos $\Leftarrow$) e, finalmente, a bi-implicação ($\Leftrightarrow$).

Os símbolos funcionais e relacionais estão enumerados em $\mathcal{S}_2$ e $\mathcal{S}_3$, respectivamente.

Em $\mathcal{C}_1$, estão as constantes correspondendo às identidades de 4 agentes em particular. Os símbolos $\boldsymbol{a}$ e $\boldsymbol{b}$ identificam dois agentes que desejam estabelecer uma chave de sessão. O símbolo $\boldsymbol{c}$ é a identidades de um agente maligno, que vai sempre tentar subverter o protocolo. A identidade $\boldsymbol{s}$ é a do o agente servidor, que geralmente fica responsável pela geração e distribuição de uma chave de sessão.

$\mathcal{C}_2$, o conjunto dos naturais, irá representar os instantes dentro de cada execução, bem como as execuções em si. O conjunto $\mathcal{C}_3$ servirá para indicar as execuções paralelas àquelas representadas pelos elementos de $\mathcal{C}_2$. Estaremos considerando a possibilidade de apenas uma execução paralela para cada execução em trânsito. Contudo, acreditamos que a linguagem proposta pode ser facilmente estendida para incluir a possibilidade de representação de infinitas execuções paralelas (através de um subscrito qualquer para os elementos de $\mathcal{C}_3$, por exemplo).

Finalmente, em $\mathcal{C}_4$, encontramos algumas mensagens em particular usadas na especificação de alguns protocolos que não os de autenticação e distribuição de chaves.

As variáveis de $\mathcal{V}_1$ e $\mathcal{V}_2$ pertencem a um mesmo tipo, que vai assumir valores no domínio dos naturais. As de $\mathcal{V}_3$ vão assumir valores no domínio das mensagens e as de $\mathcal{V}_4$ no domínio das identidades (que é um subdomínio das mensagens).

Note que estamos dispensando os símbolos que representam outros conectivos, tais como a negação (geralmente denotada por $\neg$) e a disjunção (geralmente denotada por $\vee$), bem como os símbolos $\forall$ e $\exists$ (quantificadores), isto por acreditarmos que a expressivi-

dade de nossa linguagem será suficiente para nossos propósitos, mesmo sem as sentenças que fazem uso desses conectivos. Na verdade, o quantificador $\forall$ não está presente pois assumimos que todas as variáveis de nossas fórmulas estão quantificadas universalmente. Já o quantificador $\exists$ será usado posteriormente numa extensão da linguagem criada para facilitar a representação de falhas.

Nos textos de lógica clássica, os termos formados pela aplicação de símbolos funcionais a outros termos são geralmente definidos sob a forma $F(t_1, t_2, t_3, \cdots, t_n)$ ou simplesmente $Ft_1t_2t_3\cdots t_n$ (onde F é o símbolo funcional de aridade n e $t_1, t_2, \cdots, t_n$ são termos). Como cada um dos termos de nossa linguagem tem uma grafia única, resolvemos criar uma regra de formação de termos para cada símbolo funcional.

Definição 4.3. *Considere os subconjuntos do alfabeto $\mathcal{A}$ descritos acima.* ***Termos****, cujo conjunto denotaremos por $\mathcal{MSG}$, são precisamente aquelas cadeias de $\mathcal{A}^*$ as quais podem ser obtidas por um número finito de aplicações das seguintes regras:*

- *Se $t \in \mathcal{V}_3 \cup \mathcal{V}_4 \cup \mathcal{C}_1 \cup \mathcal{C}_4$, então $\boldsymbol{t}$ é um termo.*
- *Se t é um termo, então $\boldsymbol{F(t)}$ é um termo.*
- *Se $t, s \in \mathcal{V}_4 \cup \mathcal{C}_1$, $r \in \mathcal{C}_2 \cup \mathcal{C}_3$ e $u \in \mathcal{C}_2$, então $\boldsymbol{N_t^r}$, $\boldsymbol{\overset{u}{N}_t^r}$, $\boldsymbol{K_t}$, $\boldsymbol{K_{ts}}$, $\boldsymbol{K_{ts}^r}$ e $\boldsymbol{K_t^{-1}}$ são termos.*
- *Se t e s são termos, então $\boldsymbol{\{t\}s}$ é um termo.*
- *Se t e s são termos, então $\boldsymbol{\langle t\rangle s}$ é um termo.*
- *Se t e s são termos, então $\boldsymbol{[t]s}$ é um termo.*
- *Se t e s são termos, então $\boldsymbol{t.s}$ é um termo.*

Note que o conjunto dos termos define exatamente as possíveis mensagens, daí o nome $\mathcal{MSG}$.

Das 6 primeiras regras acima obtemos as mensagens simples, e por aplicações sucessivas da última regra obtemos as mensagens compostas, ou seja, aquelas formadas pela concatenação de mensagens simples. Para tornar a definição de fórmula mais intuitiva, vamos nomear alguns conjuntos importantes:

Conjunto	**Nome**	**Significado**
$\mathcal{V}_1 \cup \mathcal{C}_2 \cup \mathcal{C}_3$	$\mathcal{EXE}$	Símbolos de execuções
$\mathcal{V}_2 \cup \mathcal{C}_2$	$\mathcal{INS}$	Símbolos de instantes
$\mathcal{V}_4 \cup \mathcal{C}_1$	$\mathcal{AGT}$	Símbolos de identidades de agentes

Pelas mesmas razões descritas acima para o caso dos termos, fomos levados a construir regras específicas para formação de cada uma de nossas fórmulas atômicas (aquelas formadas pela aplicação de um símbolo relacional a um conjunto de termos).

Definição 4.4. *Considere os subconjuntos do alfabeto $\mathcal{A}$ descritos acima.* ***Fórmulas Atômicas****, cujo conjunto denotaremos por $\mathcal{FA}$, são precisamente aquelas cadeias de $\mathcal{A}^*$ as quais podem ser obtidas por um número finito de aplicações das seguintes regras:*

- *Se $s, t \in \mathcal{INS}$, então $\boldsymbol{s < t} \in \mathcal{FA}$*
- *Se $s, t \in \mathcal{EXE}$, então $\boldsymbol{s < t} \in \mathcal{FA}$*
- *Se $s, t \in \mathcal{EXE}$, então $\boldsymbol{s || t} \in \mathcal{FA}$*
- *Se $s, t \in \mathcal{MSG}$, então $\boldsymbol{s \neq t} \in \mathcal{FA}$*
- *Se $s, t \in \mathcal{AGT}$, então $\boldsymbol{s \neq t} \in \mathcal{FA}$*
- *Se $s, t \in \mathcal{MSG}$, então $\boldsymbol{s \equiv t} \in \mathcal{FA}$*
- *Se $r \in \mathcal{EXE}$, $s \in \mathcal{INS}$, $t, u \in \mathcal{AGT}$ e $v \in \mathcal{MSG}$, então $\boldsymbol{r.s\,t \rightarrow u : v} \in \mathcal{FA}$*
- *Se $r \in \mathcal{EXE}$, $s \in \mathcal{INS}$, $t, u, v \in \mathcal{AGT}$ e $w \in \mathcal{MSG}$, então $\boldsymbol{r.s\,t_u \rightarrow v : w} \in \mathcal{FA}$*
- *Se $r \in \mathcal{EXE}$, $s \in \mathcal{INS}$, $t, u \in \mathcal{AGT}$ e $v \in \mathcal{MSG}$, então $\boldsymbol{r.s\,t \leftarrow u : v} \in \mathcal{FA}$*
- *Se $r \in \mathcal{EXE}$, $s \in \mathcal{INS}$, $t, u, v \in \mathcal{AGT}$ e $w \in \mathcal{MSG}$, então $\boldsymbol{r.s\,t \leftarrow u_v : w} \in \mathcal{FA}$*
- *Se $r \in \mathcal{EXE}$, $s \in \mathcal{INS}$, $t \in \mathcal{AGT}$ e $u \in \mathcal{MSG}$, então $\boldsymbol{r.s\,t \lhd u} \in \mathcal{FA}$*
- *Se $r \in \mathcal{EXE}$, $s \in \mathcal{INS}$, $t \in \mathcal{AGT}$ e $u \in \mathcal{MSG}$, então $\boldsymbol{r.s\,t \boxtimes u} \in \mathcal{FA}$*
- *Se $r \in \mathcal{EXE}$, $s \in \mathcal{INS}$, $t, u \in \mathcal{AGT}$ e $v \in \mathcal{MSG}$, então $\boldsymbol{r.s\,t \overset{v}{\sim} u} \in \mathcal{FA}$*
- *Se $r \in \mathcal{EXE}$, $s \in \mathcal{INS}$, $t, u \in \mathcal{AGT}$ e $v \in \mathcal{MSG}$, então $\boldsymbol{r.s\,t \overset{v}{\leftrightarrow} u} \in \mathcal{FA}$*

Para estruturar a definição de sentenças ou fórmulas, apresentamos uma definição intermediária de conjunção de formulas atômicas:

Definição 4.5. *Conjunções de fórmulas atômicas, ou simplesmente* ***conjunções****, são as cadeias de* $\mathcal{A}^*$ *obtidas através de um número finito de aplicações das seguintes regras:*

- *Se* ϕ *é uma fórmula atômica, então* $\boldsymbol{\phi}$ *é uma conjunção.*
- *Se* ϕ *e* ψ *são conjunções, então* $\boldsymbol{\phi, \psi}$ *é uma conjunção.*

Finalmente, usando a definição de conjunções acima, podemos completar a definição sintática da linguagem que será utilizada no restante deste trabalho, a qual denotaremos por $\boldsymbol{\mathcal{L}_{ep}}$:

Definição 4.6. ***Sentenças*** *ou* ***fórmulas*** *da* $\mathcal{L}_{ep}$ *são precisamente aquelas cadeias de* $\mathcal{A}^*$ *as quais podem ser obtidas por um número finito de aplicações das seguintes regras:*

- *Se* ϕ *é uma conjunção, então* $\boldsymbol{\phi}$ *é uma sentença da* $\mathcal{L}_{ep}$*.*
- *Se* ϕ *e* ψ *são conjunções, então* $\boldsymbol{\phi \Leftarrow \psi}$ *é uma sentença da* $\mathcal{L}_{ep}$*.*
- *Se* ϕ *e* ψ *são conjunções, então* $\boldsymbol{\phi \Leftrightarrow \psi}$ *é uma sentença da* $\mathcal{L}_{ep}$*.*

Especificações de protocolos e o conjunto de axiomas de nossa teoria serão escritos como conjuntos de sentenças da $\mathcal{L}_{ep}$.

Para facilitar a visualização, fórmulas do tipo $\boldsymbol{\phi, \psi}$ e dos tipos $\boldsymbol{\phi \Leftarrow \psi}$ e $\boldsymbol{\phi \Leftrightarrow \psi}$ podem também ser escritas em mais de uma linha, como nos casos abaixo:

$$\left.\begin{array}{l} E.I\ A \rightarrow S : A.B.N_A, \\ E.I\ A \boxtimes N_A, \\ E.I\ A \overset{N_A}{\sim} B \end{array}\right\} \text{Exemplo de uma fórmula da forma } \phi, \psi$$

$$\left.\begin{array}{l} E.I\ A \lhd M \Leftarrow \\ \quad E.I\ A \lhd \{M\}CH, \\ \quad E.I\ A \lhd CH \end{array}\right\} \text{Exemplo de uma fórmula da forma } \phi \Leftarrow \psi$$

$$\left.\begin{array}{l} E.I\ A \lhd M_1.M_2 \Leftrightarrow \\ \quad E.I\ A \lhd M_1, \\ \quad E.I\ A \lhd M_2 \end{array}\right\} \text{Exemplo de uma fórmula da forma } \phi \Leftrightarrow \psi$$

Inicialmente, essa definição pode parecer desnecessariamente complexa. Contudo, veremos na seção seguinte que nosso objetivo de obter sentenças de sintaxe próxima da usual e semântica intuitiva foi alcançado. Após conhecermos a semântica das relações

e funções, ficará claro que restringimos nossa linguagem àquelas sentenças que podem ser avaliadas do ponto de vista semântico, ou seja, não serão permitidas formulações de fatos absurdos, tais como mensagens enviando instantes ou chaves possuindo execuções. Contudo, nem por isso restringimos a contrução de sentenças representando aspectos da realidade tais como criptografar uma mensagem usando como chave um *nonce*, uma vez que é nesse tipo de característica, aparentemente absurda, em que pode residir uma falha não conhecida.

4.4 Semântica

Nosso universo de discurso é formado por 3 domínios distintos:

- $\mathbb{N}$ = Números naturais (que representam execuções e instantes da execução)
- $\mathbb{G}$ = Identidades de agentes
- $\mathbb{M}$ = Mensagens

Sendo $\mathbb{G}$ um subdomínio de $\mathbb{M}$, *i.e.*, $\mathbb{G} \subseteq \mathbb{M}$. O domínio $\mathbb{M}$ poderia ainda ser subdividido em *nonces*, chaves e mensagens em geral. Lembramos que não estaremos considerando essa possível subdivisão (ver 4.2). Note que estes conjuntos não são os mesmos conjuntos de *strings* definidos na seção anterior. Os símbolos do conjunto $\mathcal{MSG}$ são usados para representar os elementos do conjunto $\mathbb{M}$,que por sua vez são as reais mensagens, enquanto que os símbolos de $\mathcal{EXE}$ e $\mathcal{INS}$ representam elementos de $\mathbb{N}$, que por sua vez representam as reais execuções e instantes. Nas descrições de funções e relações a seguir, bem como nos textos que vão acompanhar as especificações de protocolos, onde se lê **mensagem M**, entenda-se **mensagem representada pelo símbolo M**. O mesmo acontecendo para identidades de agentes, execuções e instantes. Como justificado anteriormente, vamos usar uma semântica poli-sortida. Os conjuntos de variáveis $\mathcal{V}_1$ e $\mathcal{V}_2$, ou seja, variáveis de execuções e instantes, só serão considerados no domínio $\mathbb{N}$. Da mesma forma, variáveis de $\mathcal{V}_4$ só assumirão valores no domínio $\mathbb{G}$ e as de $\mathcal{V}_3$ o domínio $\mathbb{M}$. Note que, eventualmente, símbolos de $\mathcal{V}_3$ poderão assumir valores de identidades de agentes, já que trata-se de um tipo particular de mensagens. A $\mathcal{L}_{ep}$ estará completa quando apresentarmos o mapeamento de cada símbolo funcional, relacional e constante em funções, relações e elementos particulares, respectivamente, desses domínios. É importante lembrar que, no nível de análise em que estamos interessados, o real formato das mensagens do domínio $\mathbb{M}$, bem como as reais funções $\mathbb{M}^n \rightarrow \mathbb{M}$, não precisam estar completamente

definidos. Isto acontece pois estamos tratando **falhas de especificações funcionais** ou **falhas independentes de implementação** (CARLSEN, 1994a), ou seja, falhas que existem qualquer que seja a codificação das mensagens e funções de criptografia utilizadas. Algumas **falhas dependentes de implementação** também fazem parte do escopo aqui proposto. Para tratar todos os tipos de falhas dependentes de implementação, teríamos que estabelecer uma codificação para o formato das mensagens e definir por completo as funções $\mathbb{M}^n \rightarrow \mathbb{M}$. As diferentes interpretações para a $\mathcal{L}_{ep}$ serão diferenciadas apenas pela codificação usada para as mensagens e funções.

4.4.1 Semântica dos Símbolos Funcionais

O Símbolo . (ponto) (. : $\mathbb{M} \times \mathbb{M} \rightarrow \mathbb{M}$) – Vamos assumir que não há delimitadores de mensagens, ou seja, a concatenação de duas mensagens é formada pela simples justaposição da segunda à primeira. O símbolo **.** será usado para indicar essa função de concatenação. Ainda por definição, a função de concatenação é associativa, ou seja:

$$(M_1.(M_2.M_3)) \equiv ((M_1.M_2).M_3)$$

Daí a não necessidade dos parênteses.

Função de modificação de mensagens ($F : \mathbb{M} \rightarrow \mathbb{M}$) – Escrevemos $F(M)$ para indicar uma função de modificação qualquer na mensagem M, a inversão dos bits ou a adição de um valor constante. A função em si não é importante no contexto da análise de protocolos, já que ela é usada basicamente para que não haja um *replay* da mensagem. Esse tipo de função é usado geralmente da seguinte forma. Um agente emite uma mensagem (ou uma modificação dessa) criptografada e espera receber uma modificação dessa mensagem (ou a própria mensagem), criptografada, num passo posterior. Note que se não houver a modificação da mensagem, um agente maligno pode se fazer passar por outro simplesmente respondendo ao emissor exatamente a mesma mensagem que foi enviada (já que ele normalmente não possui a chave correta para criptografar a mensagem modificada). Um exemplo desta técnica está presente no protocolo Needham-Schroeder, analisado na seção 5.3.

Nonces ($N_A^E : \mathbb{N} \times \mathbb{G} \rightarrow \mathbb{M}$ e $\overset{i}{N}_A^E : \mathbb{N} \times \mathbb{N} \times \mathbb{G} \rightarrow \mathbb{M}$) – Escrevemos N_A^E para indicar o elemento de $\mathbb{M}$ que está sendo usado pelo agente A como *nonce* na execução E, ou seja, como uma mensagem gerada recentemente por ele para garantir que outras mensagens (possivelmente chaves), recebidas em passos posteriores, também sejam

recentes. Para diferenciar os vários *nonces* gerados por um agente durante uma mesma execução, usaremos um número natural (daí o domínio $\mathbb{N}$ duplicado) escrito acima do N, indicado na notação por $\boldsymbol{i}$. Sendo assim, $\overset{1}{N}{}_a^2$ e $\overset{2}{N}{}_a^2$ são dois *nonces* diferentes gerados pelo agente $\boldsymbol{a}$ na execução **2** do protocolo.

Chaves públicas ($K_A : \mathbb{G} \rightarrow \mathbb{M}$) – Para suportar a representação dos sistemas assimétricos de criptografia (ver 2.3.2), criamos as funções K e K^{-1}. A mensagem representada por K_A é a chave pública do agente A. Assumindo um sistema de criptografia assimétrico perfeito, uma mensagem criptografada com a chave pública de um agente só pode ser descriptografada pelo agente que possuir a chave privada correspondente. Isso garante a confidencialidade na comunicação. Não estamos tratando o problema da distribuição de chaves públicas (STALLINGS, 1995, p. 129) (SCHNEIER, 1996, cap. 8), portanto, vamos assumir que as chaves públicas de todos os agentes já foram corretamente distribuídas com autenticidade, lembrando que a distribuição não precisa ser realizada com confidencialidade (ver 2.6). Essa suposição estará formalmente descrita nos axiomas da seção 4.5.

Chaves privadas ($K_A^{-1} : \mathbb{G} \rightarrow \mathbb{M}$) – A mensagem representada por K_A^{-1} é a chave privada do agente A. Normalmente, a criptografia com chave privada não garante confidencialidade, sua função é garantir a autenticidade do emissor, através do método de assinatura da mensagem (ver 2.3.5). A autenticidade de uma mensagem assinada com a chave privada de um agente pode ser verificada por qualquer outro agente que possua a chave pública correspondente. Essa propriedade estará formalmente definida no conjunto de axiomas da seção 4.5. Porém, quando uma chave privada é utilizada com uma função de criptografia simétrica, as propriedades de integridade e confidencialidade estão garantidas (ver 5.5).

Chaves compartilhadas ($K_{AB} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{M}$ e $K_{AB}^E : \mathbb{N} \times \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{M}$) – Dadas duas identidades de agentes, A e B, devemos ter a possibilidade de referenciar dentre o conjunto de mensagens possíveis qual consideramos ser a chave compartilhada entre A e B. Essa mensagem é representada por K_{AB}. O $\boldsymbol{E}$ sobrescrito será usado para diferenciar as chaves de sessão, já que uma mesma chave desse tipo não deve ser utilizada em mais de uma execução de um determinado protocolo. O $\boldsymbol{E}$ será instanciado sempre com um número natural (ou um símbolo de $\mathcal{C}_2$, o qual deverá corresponder ao da execução em que tal chave será distribuída, segundo as orientações de como especificar um protocolo (ver 4.6). O fato de um agente possuir uma mensagem do tipo K_{AB} ou K_{AB}^E não implica necessariamente que tal agente sabe quais os agen-

tes que estão compartilhando tal chave. Isso pois as referidas mensagens não são marcadas com as identidades dos agentes.

Função de criptografia simétrica ($\{M\}CH : \mathbb{M} \times \mathbb{M} \rightarrow \mathbb{M}$) – Se M e CH são mensagens, escrevemos $\{M\}CH$ para representar o elemento de $\mathbb{M}$ resultado da aplicação de uma função de criptografia simétrica à mensagem M, usando-se como chave a mensagem CH. Perceba que não limitamos o termo CH ao subconjunto (de $\mathbb{M}$) das chaves, logo podemos obter expressões aparentemente estranhas, tais como uma mensagem sendo criptografada com a identidade de um agente. Contudo, estas expressões refletem bem a realidade de um ambiente hostil, e delas podem resultar falhas. A função de criptografia em si não é importante no contexto da análise de protocolos, já que, no nível em que estamos interessados, as falhas podem ocorrer independentemente de qual algoritmo de criptografia é implementado pela função. Estamos supondo que esta função garante os atributos de integridade e confidencialidade da mensagem, o que é de se esperar de qualquer função de criptografia simétrica. Essa suposição pode ser feita pois, mais uma vez, as falhas funcionais independem da qualidade da função de criptografia utilizada.

Função de descriptografia ($\langle M\rangle CH : \mathbb{M} \times \mathbb{M} \rightarrow \mathbb{G}$) – A aplicação para os símbolos $\langle$ e $\rangle$ é indicar a descriptografia de uma mensagem M utilizando-se uma mensagem CH como chave. Indicamos tal operação por $\langle M\rangle CH$. Resolvemos separar a criptografia da descriptografia para melhor tratar os casos em que as funções de criptografia simétrica não apresentam a propriedade de poderem servir também para descriptografar. Um exemplo dessa distinção pode ser encontrado na seção 5.5

Função de criptografia assimétrica ($[M]CH : \mathbb{M} \times \mathbb{M} \rightarrow \mathbb{M}$)– Sendo M e CH duas mensagens quaisquer, escrevemos $[M]CH$ para denotar a mensagem resultante da aplicação de uma função de criptografia assimétrica sobre a mensagem M, usando-se como chave a mensagem CH. Para ter acesso a mensagem M, o agente que possuir a mensagem $[M]CH$ deve também possuir a chave correspondente a CH. Quando a mensagem CH for a chave pública de um agente (K_A), vamos assumir que esse tipo de função garante a confidencialidade da mensagem M, ou seja, somente o agente que possui a chave privada correspondente pode ter acesso a mensagem M. Porém, se for usada a chave privada de um agente (K_A^{-1}), a confidencialidade se perde e ganha-se a autenticidade da mensagem M. O conceito de confidencialidade estará formalmente descrito no conjunto de axiomas da seção 4.5.

4.4.2 Semântica dos Símbolos Relacionais

Envio de mensagens ($\rightarrow$) – Escrevemos $E.I\ A \rightarrow B : M$ para indicar que, no instante I da execução E, o agente A envia para o agente B a mensagem M. Nesse caso, estamos considerando que a mensagem enviada pelo agente A tem como endereço de origem a identidade do agente A e endereço de destino a identidade do agente B. Uma abstração possível para a mensagem sendo enviada em $E.I\ a \rightarrow s : a.b.N_a$ seria:

Origem	Destino	Mensagem
a	s	$a.b.N_a$

Para o caso onde um agente tenta passar-se por outro (personificação), ou seja, o agente C envia a mensagem M para o agente B, usando como endereço de origem a identidade do agente A, escrevemos $E.I\ C_A \rightarrow B : M$. Tomaremos a identidade de um agente como um caso particular da personificação, onde o agente passa-se por ele mesmo. Dessa forma, escreveremos A_A e A, indiferentemente.

Recebimento de mensagens ($\leftarrow$) – Escrevemos $E.I\ A \leftarrow B : M$ para indicar que, no instante I da execução E, o agente A recebe do agente B a mensagem M. Nesse caso, estamos considerando que a mensagem recebida pelo agente A foi enviada pelo agente B e tem como endereço de origem a identidade do agente B. Para o caso onde um agente tenta passar-se por outro, ou seja, o agente A recebe a mensagem M, enviada na realidade pelo agente C, mas tendo como endereço de origem a identidade do agente B, escrevemos $E.I\ A \leftarrow C_B : M$

Relação de posse de mensagens ($\lhd$) – De acordo com a divisão da relação de conhecimento proposta por Carlsen (1994b, p. 40), estamos considerando essa relação como a posse da mensagem, ou seja, dizemos que um agente possui uma mensagem quando esse tem em seu poder a real cadeia de bits que representa a mensagem. Não nos interessa, nem precisamos representar, nada a respeito do significado que os agentes dão às mensagens. Isso por acreditarmos que a posse como aqui considerada já é, por si só, um indício de falha, a ser tratado como uma outra falha qualquer. A única exceção está na relação $\leftrightarrow$, descrita a seguir. Para indicar que num instante I, da execução E, o agente A possui a mensagem M, escrevemos $E.I\ A \lhd M$.

Armazenamento de mensagens ($\boxtimes$) – Ao criar ou receber uma mensagem, um agente pode desejar armazená-la para uso posterior. Para indicar que no instante I da execução E o agente A armazena a mensagem M, escrevemos $E.I\ A \boxtimes M$. Estamos

considerando um sistema de armazenamento onde uma mensagem armazenada por um agente não é visível a outro, e onde é sempre possível recuperar uma mensagem armazenada.

Relação de ordem entre os instantes e entre execuções ($<$) – Se uma ação ocorre num instante I_1 anterior a outro instante I_2 dentro de uma mesma execução de um protocolo, então, necessariamente, $I_1 < I_2$. Já que os instantes das execuções têm seu domínio em $\mathbb{N}$, essa relação é idêntica à relação menor que da aritmética. O conjunto de axiomas do ambiente de comunicação (ver 4.5), contém a sentença que permite a dedução de fatos do tipo $1 < 2$, $3 < 5$ etc. Da mesma forma, $E_1 < E_2$ indica que todos os passos da execução E_2 acontecem após o encerramento da execução E_1.

Chave boa para comunicação ($\leftrightarrow$) – O principal objetivo de alguns protocolos de que estamos tratando é estabelecer uma chave de sessão entre dois agentes. Essa chave normalmente deve ser distribuída com confidencialidade, autenticidade e integridade, por um terceiro agente, confiável aos dois primeiros. Para formalizar o conceito onde, numa execução E no instante I, um agente A considera uma chave CH boa para ser usada na comunicação com outro agente B, usamos a relação $E.I\ A \overset{CH}{\leftrightarrow} B$. Só consideraremos chaves compartilhadas entre dois agentes. É função do projetista do protocolo estabelecer, ou seja, expressar através de uma sentença, o instante em que um agente considera uma chave boa para comunicação com outro agente. No tocante aos componentes do conhecimento, as fórmulas da $\mathcal{L}_{ep}$ representam apenas posse de mensagens (através da relação $\lhd$). A relação $\leftrightarrow$ é a única que representa a atribuição de um significado a uma mensagem por parte de um agente.

Relacionamento entre identidade e *nonce* ($\sim$) – A fim de garantir a autenticidade de uma chave de sessão, os agentes devem relacionar seus *nonces* aos agentes com quem desejam estabelecer tal chave. Esse relacionamento pode ser entendido como o simples armazenamento da informação de que um determinado *nonce* está ligado (relacionado) a um determinado agente. Assim, ao receber futuramente uma chave de sessão, o agente pode, além de conferir o frescor da chave através do *nonce*, saber também com qual agente tal chave deve ser usada. Também é função do projetista descrever explicitamente com qual agente tal mensagem está relacionada, sob pena de permitir uma falha de autenticidade caso não o faça (ver 5.3).

Desigualdade entre mensagens ($\neq$) – Usamos $M_1 \neq M_2$ para indicar que as mensagens M_1 e M_2 são diferentes, ou seja, referem-se a elementos distintos de $\mathbb{M}$.

Para denotar a igualdade entre mensagens numa mesma sentença, basta usarmos a mesma variável para referenciá-las. Dentre os axiomas do ambiente de comunicação (ver 4.5), está um que permite a dedução de instâncias dessa relação.

Paralelismo entre execuções ($||$) – Esta relação será usada para indicar que duas execuções de um mesmo protocolo, ou até de protocolos distintos, têm seus passos intercalados antes de alguma delas ter chegado ao fim. Escreveremos $E_1||E_2$, por exemplo, para indicar que as execuções E_1 e E_2 são paralelas. Admitiremos a possibilidade de apenas uma execução paralela. O conjunto de constantes $\mathcal{C}_3$ contém as representações de todas as possíveis execuções paralelas, cada uma correspondendo a uma execução em trânsito, a qual, por sua vez, é indicada pelos elementos de $\mathcal{C}_2$. No conjunto de axiomas proposto a seguir, essa correspondência estará formalizada.

Equivalência entre Mensagens ($\equiv$) – Algumas funções de criptografia e de modificação possuem propriedades singulares e, quando combinadas de uma maneira particular, têm seus efeitos anulados, não alterando o conteúdo da mensagem original. Para indicar a equivalência entre mensagens que sofreram, ou não, tais transformações, usaremos o símbolo $\equiv$. Por exemplo, uma mensagem criptografada com uma chave privada e descriptografada com essa mesma chave é equivalente à mensagem original, de antes da aplicação da criptografia. Por exemplo, para expressar essa propriedade em particular, escrevemos: $\langle\{M\}CH\rangle CH \equiv M$. Como veremos, essa propriedade também estará formalizada no conjunto de axiomas proposto a seguir.

4.4.3 Semântica das Constantes

Constantes que representam identidade de agentes (a, b, c, s) – Essas constantes são mapeadas no domínio $\mathbb{G}$. No caso dos protocolos de autenticação e distribuição de chaves, as constantes $\boldsymbol{a}$ e $\boldsymbol{b}$ representam os agentes que desejam estabelecer uma chave de seção. O símbolo $\boldsymbol{c}$ representa o agente maligno, o qual tenta subverter o protocolo. O agente $\boldsymbol{s}$ é aquele com quem $\boldsymbol{a}$ e $\boldsymbol{b}$ inicialmente compartilham chaves e confiam para gerar a chave de sessão. Para outros tipo de protocolo, os agentes $\boldsymbol{a}$ e $\boldsymbol{b}$ podem ser considerados simplesmente como o iniciador e o respondedor, respectivamente.

Informação a ser transmitida (*info*) – Alguns protocolos têm como objetivo transmitir uma informação conhecida por um agente para outro. Pode-se desejar ainda

que tal transmissão seja feita com confidencialidade, integridade e, opcionalmente, autenticidade. Para representar de forma geral esse tipo de mensagem, usaremos a constante ***info***.

Mensagem sem significado (*lixo*) – A constante *lixo* denota uma mensagem sem significado, transmitida geralmente apenas para complementar uma execução onde há uma subversão do protocolo e os agentes malignos não desejem levantar suspeitas sobre sua presença.

4.5 Axiomas

A partir desta seção, as fórmulas da $\mathcal{L}_{ep}$ serão rotuladas para posterior referência. Tais rótulos estarão à esquerda das fórmulas, e serão siglas para expressões que retratam os axiomas. As expressões usadas estarão ao lado dos axiomas, entre parênteses, juntamente com um rápido comentário, a fim de facilitar futuras consultas ao conjunto de axiomas.

O conjunto de axiomas a ser apresentado torna explícito o conhecimento tácito sobre protocolos de criptografia e o ambiente em que esses estão inseridos. Com eles (e um método de inferência) vai ser possível formalizar o raciocínio que leva à dedução de falhas.

Como estamos interessados em analisar protocolos de segurança, vamos considerar o ambiente como o mais hostil possível, ou seja, sempre que possível, vamos permitir a dedução do caso onde ocorre uma falha de segurança. Com esse princípio em mente, podemos formular o conjunto de axiomas da teoria proposta, o qual denotaremos por Δ_{ep}. Lembramos que todos os axiomas são sentenças da $\mathcal{L}_{ep}$, ou seja, $\Delta_{ep} \subset \mathcal{L}_{ep}$. Os axiomas são apresentados nas seções seguintes, onde estão agrupados de acordo com seus significados.

4.5.1 Transmissão e Recepção de Mensagens

De acordo com o definido na semântica dos símbolos de envio e recebimento, podemos interpretar o caso onde não há personificação como um caso particular, onde um agente passa-se por ele mesmo. Sendo assim, onde se lê A, entenda-se como A_A, e vice-versa. Os axiomas a seguir estarão definidos para o caso geral da personificação, mas com a convenção acima assumida, podem ser aplicados no caso onde não há personificação.

Se um agente A envia para o agente B uma mensagem M num instante I_1 qualquer, é possível que, num instante I_2 posterior, a mensagem M chegue a seu destino inalterada.

Contudo, dentro de nosso princípio de considerar também o pior caso, é razoável assumir que o canal de transmissão é visível a todos os agentes, ou seja, todos recebem a mensagem M num instante posterior. Formalmente escrevemos:

CV	$E.I_2\, A \leftarrow C_S : M \Leftarrow$ $E.I_1\, C_S \rightarrow B : M,$ $I_1 < I_2$	O canal é visível a todos, mesmo na personificação. (**C**anal **V**isível)

Lembrando que durante o restante desta seção, as sentenças de cada axioma serão apresentadas como acima, sendo a sigla à sua esquerda, em negrito, usada para futuras referências, e o comentário à sua direita servindo como como breve explanação da característica modelada pelo axioma.

Obviamente, dentre todos os agentes está o agente de destino original. A importância desse axioma está em permitir a dedução da possibilidade de um outro agente, que não o destinatário original, receber a mensagem enviada. É dessa possibilidade que pode surgir uma falha.

Um caso particular do axioma acima é o da **transmissão normal**, ou seja, aquela onde a mensagem chega inalterada a seu endereço de destino original, ou formalmente:

TN	$E.I_2\, B \leftarrow C_A : M \Leftarrow$ $E.I_1\, C_A \rightarrow B : M,$ $I_1 < I_2$	Transmissão normal da mensagem. Neste caso, a mensagem chega a seu destino original. (**T**ransmissão **N**ormal)

Para que a personificação ocorra, basta que o agente que deseja passar-se por outro possua a mensagem que deseja enviar, já que tudo que resta fazer é inserir como endereço de origem o endereço desejado:

PER	$E.I\, C_A \rightarrow B : M \Leftarrow$ $E.I\, C \lhd M$	O agente C passa-se por A. (**PER**sonificação)

A aparente restrição do envio só acontecer no instante da posse da mensagem será superada quando apresentarmos o axioma que estende a posse aos instantes posteriores ao que ela foi adquirida.

Estendendo o conceito da personificação, é razoável assumir que o agente que recebe a mensagem personificada é ludibriado quanto à sua origem:

RIT	$E.I\,B \leftarrow A : M \Leftarrow$ $E.I\,C_A \rightarrow B : M$	Para os efeitos da análise de vulnerabilidades, o agente B recebe a mensagem como se fosse vinda do agente A. (**R**ecebe com a **I**dentidade **T**rocada)

4.5.2 Suposições Iniciais

Todos os agentes possuem seus respectivos *nonces* em qualquer instante de qualquer execução:

PNC	$E_1.I\,A \lhd N_A^{E_2}$ ou $E_1.I\,A \lhd \overset{i}{N}_A^{E_2}$	Os agentes possuem seus *nonces* a qualquer instante de qualquer execução. (**P**ossui ***N**on**C**es*)

O rótulo PNC será utilizado para ambos os casos de posse de *nonces* acima. Deverá estar claro no contexto a que sentença estaremos nos referindo. Essa observação também será válida para os demais axiomas definidos com mais de uma sintaxe.

Todos os agentes conhecem as identidades de todos os outros em qualquer instante de qualquer execução:

PID	$E.I\,A \lhd B$	Os agentes se conhecem, ou seja, têm a seu dispor as identidades dos outros agentes. (**P**ossui **ID**entidades)

4.5.3 Posse de Mensagens

Ao receber uma mensagem, um agente passa imediatamente a possuí-la:

PR	$E.I\,A \lhd M \Leftarrow$ $E.I\,A \leftarrow C_B : M$	(**P**ossui o que **R**ecebe)

Se um agente possui uma mensagem num instante, não há porque essa posse seja perdida, ou seja, nos instantes posteriores o agente continua possuindo o que possuía anteriormente. Isso pode ser formulado como:

PIP	$E.I_2\, A \lhd M \Leftarrow$ $E.I_1\, A \lhd M,$ $I_1 < I_2$	Se possuía, continua a possuir. (**P**ossui num **I**nstante **P**osterior)

A posse também deve ser estendida às diferentes execuções onde foi adquirida. Note que, dentro de uma mesma execução (caso acima), basta que o instante em questão seja posterior ao instante em que o agente já conhecia a mensagem. Já entre execuções diferentes, podemos considerar que a mensagem é conhecida em qualquer instante, desde que trate-se de uma execução posterior ou paralela. Os axiomas abaixo formalizam essas duas possibilidades, respectivamente:

PEP	$E_1.I_1\, A \lhd M \Leftarrow$ $E_2.I_2\, A \lhd M,$ $E_2 < E_1$	Se possuía numa execução anterior, então continua a possuir em execuções posteriores. (**P**ossui numa **E**xecução **P**osterior)

PEL	$E_1.I_1\, A \lhd M \Leftarrow$ $E_2.I_2\, A \lhd M,$ $E_1 \| \| E_2$	Se possuía numa execução, então possui numa execução paralela. (**P**ossui numa **E**xecução Para**L**ela)

Apesar de não haver delimitadores entre as submensagens de uma mensagem composta, é razoável assumir que todos os agentes conhecem o formato das mensagens trocadas no protocolo. Logo, se um agente possui uma mensagem composta, obviamente possui as mensagens simples que a compõem. Contudo, se um agente recebe uma mensagem num formato diferente do esperado, sua interpretação da mensagem continua seguindo as regras do protocolo, o que pode vir a provocar um resultado inesperado. A posse das submensagens que formam uma mensagem composta pode ser formulada como:

MS	$E.I\, A \lhd M_1.M_2 \Leftrightarrow$ $E.I\, A \lhd M_1,$ $E.I\, A \lhd M_2$	Se possui o todo, possui as partes. Note a bi-implicação. (**M**ensagem **S**imples)

O fato de um agente possuir uma mensagem criptografada não garante que ele possua seu conteúdo. Para isso, é necessário que ele possua também a chave usada na criptografia:

MCC	$E.I\ A \lhd M \Leftarrow$	Se possui a chave, possui o conteúdo.
	$E.I\ A \lhd \{M\}CH$,	(**M**ensagem criptografada com **C**have
	$E.I\ A \lhd CH$	**C**ompartilhada)

Note que o agente vem a possuir a mensagem no mesmo instante que possui a mensagem criptografada e a chave, ou seja, estamos considerando que a operação de descriptografia é instantânea.

Um agente é capaz de modificar uma mensagem que ele possui através da função de modificação de mensagens (F):

PMM	$E.I\ A \lhd F(M) \Leftarrow$	Se possui a mensagem, possui sua
	$E.I\ A \lhd M$	modificação. (**P**ossui **M**ensagem
		Modificada)

Um agente pode criptografar uma mensagem usando como chave uma outra mensagem (que pode eventualmente ser a mesma), desde que ele possua ambas as mensagens:

PMC	$E.I\ A \lhd \{M\}CH \Leftarrow$	Se possui uma mensagem e uma chave
	$E.I\ A \lhd M$,	(que pode ser uma mensagem
	$E.I\ A \lhd CH$	qualquer), então possui a mensagem
		criptografada. (**P**ossui **M**ensagem
		Criptografada)

Chaves de sessão não devem ser reutilizadas pois, dados tempo e recursos suficientes aos agentes malignos, nenhum algoritmo de criptografia pode ser considerado seguro. Vamos considerar que as chaves compartilhadas permanentes possuem um tamanho tal que inviabiliza um ataque por força bruta. Porém, para o caso das chaves de sessão, vamos considerar que, uma vez que elas tenham sido utilizadas numa sessão (execução), outros agentes, que não os que compartilham a chave, vão ter acesso a ela em outras execuções posteriores, desde que tenham tido acesso (posse) de uma mensagem qualquer criptografada com a referida chave de sessão:

TI	$E_1.I_1\ A \lhd K_{ab}^{E1} \Leftarrow$	(**T**empo **I**nfinito)
	$E_2.I_2\ A \lhd \{M\}K_{ab}^{E2}$,	
	$E_2 < E_1$	

Não vamos tratar o problema da distribuição de chaves públicas. Vamos assumir que tais chaves já foram distribuídas com autenticidade, ou seja, todos os agentes possuem as chaves públicas de todos os outros, inclusive a sua própria:

PCB	$E.I\ A \lhd K_B$	Todo agente possui as chaves públicas de todos os demais agentes. (**P**ossui **C**have pú**B**lica)

Para que o sistema de criptografia assimétrico funcione corretamente, cada agente deve ter em seu poder sua chave privada:

PCP	$E.I\ A \lhd K_A^{-1}$	Todo agente possui sua chave privada. (**P**ossui **C**have **P**rivada)

Uma mensagem criptografada com a chave pública de um agente pode ser descriptografada por quem possuir a chave privada correspondente. No início da execução, é de se esperar que somente o dono da chave privada realmente a possua. Porém, no desenrolar da execução, nada garante que essa situação vai permanecer estável. Ou seja, por algum motivo, um agente pode vir a possuir a chave privada de outro. O axioma abaixo vai permitir a dedução do fato desse novo possuidor ter acesso aos dados contidos nas mensagens criptografadas com a chave pública do primeiro:

MCB	$E.I\ A \lhd M \Leftarrow$ $E.I\ A \lhd [M]K_B,$ $E.I\ A \lhd K_B^{-1}$	Se possui a chave privada correspondente, então possui a mensagem. (**M**ensagem **C**riptografada com chave pú**B**lica)

Um agente criptografa uma mensagem com sua chave privada para criar uma assinatura digital (ver 2.3.5). Essa assinatura vai permitir que outros agentes verifiquem a autenticidade das mensagens criadas ou emitidas por tal agente. A assinatura digital não garante a confidencialidade da mensagem em questão. Isso implica dizer que uma mensagem criptografada com a chave pública de um agente (assinada) é visível a todos os agentes que possuem a chave pública correspondente:

MA	$E.I\ A \lhd M \Leftarrow$ $E.I\ A \lhd [M]K_B^{-1},$ $E.I\ A \lhd K_B$	Se possui a chave pública correspondente, então possui a mensagem. (**M**ensagem **A**ssinada)

4.5.4 Axiomas Auxiliares

A simetria das funções de criptografia pode ser formalmente representada por:

MD $\langle\{M\}CH\rangle CH \equiv M$ (**M**ensagem **D**escriptografada com mesma chave)

Qualquer sentença que contenha uma mensagem M pode ser reescrita usando-se uma mensagem equivalente a M. Para não termos de exaurir todas as possibilidades, vamos definir um modelo de axioma, ao qual poderemos corresponder diversas sentenças. Sendo φ uma sentença contendo a mensagem M_1 e ψ essa mesma sentença tendo a mensagem M_1 substituída por M_2, temos:

SME $\psi \Leftarrow \varphi, M_1 \equiv M_2$ (**S**ubstituição por **M**ensagem **E**quivalente)

As duas sentenças abaixo são exemplos da "instanciação" desse modelo de axioma:

$$E.I\ A \rightarrow B : M_2 \Leftarrow E.I\ A \rightarrow B : M_1,\ M_2 \equiv M_1$$

$$E.I\ C \lhd M_2 \Leftarrow E.I\ C \lhd M_1,\ M_2 \equiv M_1$$

Apesar de não estarmos considerando tipos distintos para cada subclasse de mensagem bem conhecida (chaves, identidades, *nonces* etc), propomos a criação de alguns axiomas para formalizar a diferença entre alguns casos particulares de mensagens. Esse tipo de axioma será útil para formalizar falhas decorrentes do uso de mensagens de forma irregular.

A diferença entre mensagens compostas pode ser expressa por:

MDF $M_1.M_2 \neq M_3.M_2 \Leftarrow M_1 \neq M_3$ ou $M_1.M_2 \neq M_1.M_3 \Leftarrow M_2 \neq M_3$ ou $M_1.M_2 \neq M_3 \Leftarrow M_1 \neq M_3$ Mensagens compostas são diferentes se alguma de suas mensagens simples for diferente. (**M**ensagens **D**i**F**erentes)

Não vamos considerar a possibilidade de dois ou mais agentes possuírem a mesma identidade, ou seja, as identidades dos agentes são únicas. Neste axioma, vamos recorrer a uma definição intuitiva de diferença. Vamos considerar que duas identidades são diferentes quando forem representadas por duas constantes diferentes, ou seja, sempre que a instância de A for diferente da instância de B, escreveremos:

IU	$A \neq B \Leftarrow$ A instância de A for diferente da de B.	Não existem dois agentes com a mesma identidade. (**I**dentidade **Ú**nica)

A partir desse axioma, poderemos derivar sentenças do tipo $a \neq b$, $c \neq s$ etc.

A diferença entre mensagens pode ser expressa também em função das identidades dos agentes:

DCB	$K_A \neq K_B \Leftarrow A \neq B$	(**D**iferença entre **C**haves pú**B**licas)
DCC	$K_{AB}^{E_1} \neq K_{CS}^{E_2} \Leftarrow A \neq C$ ou $K_{AB}^{E_1} \neq K_{CS}^{E_2} \Leftarrow B \neq S$ ou $K_{AB}^{E_1} \neq K_{CS}^{E_2} \Leftarrow E_1 < E_2$ ou $K_{AB}^{E_1} \neq K_{CS}^{E_2} \Leftarrow E_2 < E_1$	(**D**iferença entre **C**haves **C**ompartilhadas)
DCP	$K_A^{-1} \neq K_B^{-1} \Leftarrow A \neq B$	(**D**iferença entre **C**haves **P**rivadas)

Também consideraremos chaves diferentes de *nonces* e de identidades, bem como *nonces* diferentes de identidades, e ainda a diferença entre os vários tipos de chaves (compartilhadas, públicas e privadas):

DTM	$K_A \neq N_B^E$ ou $K_A \neq \overset{i}{N}_B^E$ ou $K_A \neq B$ ou $N_A^E \neq B$ ou $K_A \neq K_{BC}$ ou $K_A \neq K_{BC}^E$ ou $K_A \neq K_B^{-1}$ ou $K_A^{-1} \neq K_{BC}$ ou $K_A^{-1} \neq K_{BC}^E$ ou $N_A^E \neq K_{BC}$ ou $\overset{i}{N}_A^E \neq K_{BC}$ ou $N_A^{E_1} \neq K_{BC}^{E_2}$ ou $\overset{i}{N}_A^{E_1} \neq K_{BC}^{E_2}$ ou $N_A^E \neq K_{BC}$ ou $K_A^{-1} \neq N_B^E$ ou $K_A^{-1} \neq \overset{i}{N}_B^E$	Chaves são diferentes de *nonces* e de identidades de agentes. Identidades são diferentes de *nonces*. (**D**iferença entre **T**ipos **M**ensagens)

Por fim, a diferença entre mensagens será considerada comutativa:

CMD $M_1 \neq M_2 \Leftarrow M_2 \neq M_1$ — A direfença entre mensagens é comutativa. (**C**omutatividade entre **M**ensagens **D**iferentes)

De forma análoga à diferença entre as identidades dos agentes, vamos axiomatizar a relação de ordem entre instantes e entre execuções recorrendo aos conceitos bem conhecidos da relação "menor que" da aritmética:

MQ $I_1 < I_2 \Leftarrow$ (**M**enor **Q**ue)
O número natural representado por I_1 for menor que aquele representado por I_2.

Para cada execução em trâsito, podemos analisar uma execução paralela correspondente:

PE $E_1||E_1^{||}$ (**P**aralelismo entre **E**xecuções)
ou
$E_1^{||}||E_1$

Todos os agentes possuem a mensagem *lixo*, podendo portando enviá-la a qualquer instante:

PLX $E.I\ A \lhd lixo$ (**P**ossui **LiX**o)

4.6 Como Especificar um Protocolo

Qualquer processo de especificação implica numa passagem do informal para o formal, sendo portanto, inerentemente sujeito a um entendimento particular da realidade pelo projetista. A semântica da $\mathcal{L}_{ep}$, apresentada anteriormente, mostra o mapeamento entre aspectos do mundo real e as sentenças correspondentes em nossa linguagem. Para tornar a tarefa de especificação ainda mais intuitiva, vamos oferecer orientações sobre como passar de uma especificação em notação padrão (CARLSEN, 1994c) para uma especificação em $\mathcal{L}_{ep}$. Como ficará evidenciado, a notação padrão é incompleta sob vários aspectos. Portanto, esse mapeamento também está sujeito a um entendimento particular da notação padrão. Os trechos sublinhados nas sentenças desta seção realçam a parte da sintaxe

criada para tratar (especificar) o aspecto que estiver sendo discutido.

4.6.1 Especificando Explicitamente o Tempo

Na notação padrão, os passos do protocolo são numerados, sugerindo assim a ordem em que devem ser executados. O uso de constantes (os símbolos 1,2,3...) implica na não possibilidade de especificarmos passos intermediários entre dois passos consecutivos. Para exemplificar essa questão, considere o trecho de protocolo abaixo:

$$1.\, A \rightarrow B : \{M\}K_a$$
$$2.\, B \rightarrow A : \{\{M\}K_a\}K_b$$

A primeira sentença indica uma ação (envio de mensagem) a ser tomada no primeiro passo. A segunda sentença representa outra ação (outro envio de mensagem) a ser executada no segundo passo. A questão é: como representar o instante em que ações intermediárias, tais como o recebimento da primeira mensagem ou a aplicação da função de criptografia, são executadas? Para fugir dessa limitação, escolhemos representar os instantes de uma execução através de variáveis. Usamos ainda uma variável (E) para representar a execução do protocolo, podendo assim diferenciá-la de outras execuções, sejam elas simultâneas, anteriores ou posteriores. Tal variável não tem correspondente na notação padrão. Para especificar o primeiro passo acima em $\mathcal{L}_{ep}$, escrevemos:

$$\underline{E.I}\; A \rightarrow B : \{M\}K_A$$

A notação padrão também não fornece meios para especificarmos as relações de dependência de tempo entre ações. No caso acima, não temos como especificar que o segundo passo do protocolo só pode acontecer depois do primeiro (na verdade, após uma conseqüência do primeiro, no caso, o recebimento da mensagem M por B). Para especificar esse tipo de dependência, escrevemos, para o caso acima:

$$E.I\; B \rightarrow A : \{M\}K_B \Leftarrow$$
$$E.I_2\; B \leftarrow A : M$$
$$\underline{I_2 < I}$$

Ou seja, caso o agente B receba uma mensagem M num instante I_2 qualquer, esse mesmo agente, num instante I posterior, responde com essa mesma mensagem criptografada com sua chave pública K_B. O uso da variável M em vez da mensagem $\{M\}K_A$ será explicado na seção 4.6.5.

4.6.2 Especificando as Suposições Iniciais

Um dos pontos negativos da notação padrão é sua falta de habilidade para especificar suposições iniciais (ver 2.5). Portanto, não há como mapear sentenças desse tipo para a $\mathcal{L}_{ep}$. Os axiomas PNC e PID são suposições iniciais válidas para qualquer protocolo, portanto não precisamos reescrever nenhuma sentença a respeito de agentes possuindo seus próprios *nonces* ou identidades de outros agentes. O tipo de suposição inicial mais comum está relacionado com a posse das chaves compartilhadas. Para indicar, por exemplo, que os agentes $\boldsymbol{a}$ e $\boldsymbol{s}$ compartilham uma chave K_{as} em qualquer instante de qualquer execução, escrevemos as 2 sentenças seguintes:

$$E.I\ a \lhd K_{as}$$
$$E.I\ s \lhd K_{as}$$

lembrando que a relação $\lhd$ garante que o agente $\boldsymbol{a}$ possui a mensagem K_{as} e sabe que trata-se de uma chave para comunicação privada com $\boldsymbol{s}$. Outra suposição comum nos protocolos de autenticação e distribuição de chaves é a de que um agente confiável a todos os outros é responsável pela criação das chaves de sessão. Para efeito de análise, esse fato é análogo ao de que esse "terceiro confiável" conhece antecipadamente todas as chaves de sessão entre todos os outros agentes. Para especificar, por exemplo, que o agente $\boldsymbol{s}$ é responsável pela chave de sessão a ser usada entre $\boldsymbol{a}$ e $\boldsymbol{b}$, escrevemos:

$$E.I\ s \lhd K_{ab}^{E}$$

4.6.3 Especificando o Passo Inicial

Por não diferenciar constantes de variáveis, uma outra falha da notação padrão é dar a entender que o primeiro passo de um protocolo só pode ser dado por um agente em particular. Essa restrição parece não corresponder à realidade, uma vez que outros agentes podem possuir as mesmas capacidades e interesses, inclusive o agente maligno $\boldsymbol{c}$. Muitas vezes, o primeiro passo de um protocolo não está condicionado a nenhum conhecimento específico por parte do agente que vai iniciar a execução. Nesse caso, basta especificar o formato da mensagem a ser enviada. Vamos supor que queremos especificar que, para dar o pontapé inicial, um agente qualquer precisa apenas enviar para o terceiro confiável sua identidade, a identidade do segundo agente (com o qual ele deseja estabelecer uma chave de sessão) e um *nonce* gerado por ele. Na notação padrão, isso é usualmente descrito

como:

$$1.\, A \rightarrow S : A, B, N_a$$

Perceba que são usados símbolos de uma mesma classe (letras maiúsculas) quando se deseja representar dois conceitos diferentes. Para o tipo de passo inicial descrito acima, A deveria representar um agente qualquer, e S um agente em particular. De maneira análoga, o *nonce* representado por N_a não parece ter relação com o agente emissor A, quando na verdade deveria. Para representar essas diferenças, escrevemos em $\mathcal{L}_{ep}$:

$$E.I\ \underline{A} \rightarrow \underline{s} : A.B.N^{E}_{\underline{A}}$$

Note o uso da variável A e da constante $\boldsymbol{s}$. Uma vez que *nonces* possuem como principal função garantir o frescor das mensagens trocadas numa execução, o projetista deve também especificar que o nonce gerado é específico para a execução em curso. Isso deve ser feito incluindo-se o símbolo da execução E sobrescrito ao símbolo funcional N, como mostra o exemplo acima.

Nos casos onde são enviadas mensagens criptografadas ou mensagens que nem todos os agentes possuem antecipadamente, formulamos sentenças do tipo **envio. . . se condições**. No exemplo acima, se a identidade do segundo agente tivesse que ser criptografada, juntamente com o *nonce* do primeiro agente, usando-se a chave compartilhada entre o primeiro agente e o agente confiável, teríamos uma sentença do tipo:

$$\begin{aligned} &E.I\ A \rightarrow s : A.\{B.N^{E}_{A}\}K_{As} \Leftarrow \\ &E.I\ \underline{A} \lhd K_{\underline{A}s} \end{aligned}$$

Condicionando a execução dessa ação aos agentes que possuem uma chave compartilhada com $\boldsymbol{s}$.

4.6.4 Especificando Chaves de Sessão

Uma vez que estaremos considerando a possibilidade de uma execução paralela para cada execução em trânsito, bem como a ocorrência de diversas execuções em seqüência, torna-se imprescindível a diferenciação entre as chaves de sessão de cada execução. Para tal, utilizaremos um sobrescrito nas chaves de sessão. Na especificação de um protocolo, o projetista deverá deixar claro que, a cada execução, uma nova chave de sessão é gerada por um dos agentes (geralmente pelo terceiro confiável). Para isso, sugerimos utilizar como sobrescrito ao símbolo da chave de sessão o mesmo símbolo que representa a execução em

trânsito do protocolo, geralmente denotado por $\boldsymbol{E}$. Por exemplo, para especificar que um agente, (possivelmente o terceiro confiável) incluí na resposta à uma mensagem recebida uma chave de sessão a ser utilizada por dois outros agentes (identificados na mensagem recebida), escrevemos:

$$\begin{aligned}&\underline{E}.I\ S \rightarrow AG_1 : \{AG_1.AG_2.K^{\underline{E}}_{AG_1\,AG_2}\} \Leftarrow \\ &\quad \underline{E}.I_2\ S \leftarrow AG_1 : AG_1.AG_2, \\ &\quad I_2 < I\end{aligned}$$

Exemplos reais desse tipo de especificação estão descritos no capítulo 5.

4.6.5 Especificando a Dependência entre Envio e Recebimento

Da maneira como as ações são especificadas na notação padrão, não fica clara a dependência entre o envio e o recebimento de mensagens. Pior ainda, a dependência do formato da mensagem enviada com relação à mensagem recebida também não está especificada. Considere o trecho inicial do protocolo Otway-Rees, citado em Boyd e Mao (1993) e descrito em notação padrão como:

$$\begin{aligned}&1.\ A \rightarrow B : M, A, B, \{N_a, M, A, B\}K_{as} \\ &2.\ B \rightarrow S : M.A.B.\{N_a, M, A, B\}K_{as}.\{N_B.M.A.B\}K_{bs}\end{aligned}$$

Qualquer conhecedor do protocolo sabe que a mensagem enviada pelo agente B no segundo passo não tem sempre o formato apresentado. A primeira parte dessa mensagem deve ser exatamente a mesma mensagem recebida por B, que por sua vez pode ou não (no caso de uma tentativa de ataque) ser a mesma enviada pelo agente A no passo 1. Da forma que foi apresentada, entende-se que somente o primeiro caso pode ser verdade. A especificação que propomos para expressar esse tipo de dependência é formada por duas sentenças, uma para cada passo da especificação em NP descrita acima:

$$E.I\ A \rightarrow B : N^1_A.A.B.\{N^2_A.N^1_A.A.B\}K_{As}$$

como sentença para o primeiro passo, e:

$$\begin{aligned}&E.I\ B \rightarrow s : \underline{M_1}.\underline{AG_1}.\underline{AG_2}.\underline{M_2}.\{N_B.\underline{M_1}.\underline{AG_1}.\underline{AG_2}\}K_{Bs} \Leftarrow \\ &\quad E.I_2\ B \leftarrow A : \underline{M_1}.\underline{AG_1}.\underline{AG_2}.\underline{M_2}, \\ &\quad I_2 < I\end{aligned}$$

representando o segundo passo.

Vemos agora que o comportamento do agente B pode ser diferente daquele aparentemente expresso pela notação padrão. A escolha de qual chave usar na criptografia da mensagem

$$\{N_B.M_1.AG_1.AG_2\}K_{Bs}$$

bem como o próprio conteúdo dessa mensagem, são dependentes dos valores das submensagens recebidas por B. Já a identidade do agente de destino não está condicionada a nenhuma submensagem recebida por B, sendo sempre igual à do terceiro confiável. Esse fato, que também não estava corretamente especificado na notação padrão, agora fica evidenciado pelo uso da constante $\boldsymbol{s}$.

4.6.6 Especificando as Ações Internas

Falhas devidas a falta de especificação das ações internas são descritas no capítulo 2. Vamos considerar o passo inicial do protocolo Needham-Schroeder, tipicamente descrito em notação padrão como (ver 2.5):

$$1.\, A \rightarrow S : A, B, N_a$$

O motivo pelo qual o agente A envia o *nonce* N_a é o de poder garantir que uma submensagem recebida no futuro, e que esteja concatenada com tal *nonce* e criptografada com uma chave conhecida por A, tenha sido criada na execução corrente do protocolo e não numa execução diferente (possivelmente anterior). Contudo, para que esse esquema funcione, é preciso que o agente A armazene o *nonce* enviado para posterior conferência com um valor recebido e supostamente igual ao do *nonce* inicial. A ação de armazenamento do *nonce* não está especificada explicitamente na sentença acima. O melhor que se pode obter de uma notação como essa é uma convenção de que todos os *nonces* enviados são previamente armazenados. Esse mecanismo poderia ser incorporado à $\mathcal{L}_{ep}$ na forma de um axioma que permitisse a dedução do armazenamento a partir do envio. Contudo, preferimos não adicionar tal axioma e forçar que o autor do protocolo especifique explicitamente esse tipo de ação. Acreditamos que uma especificação mais rica ajudaria, não só o autor a organizar melhor suas idéias, mas, principalmente, o implementador do protocolo a não cometer o erro de não codificar as verificações representadas pelas ações internas. A $\mathcal{L}_{ep}$ permite a especificação do armazenamento através do uso da relação $\boxtimes$, a qual normalmente estará vinculada ao envio do *nonce* sendo armazenado:

$$E.I\ A \rightarrow s : A.B.N_A^E, \quad E.I\ A \underline{\boxtimes}\ N_A^E$$

Outra ação interna normalmente não especificada acontece quando um agente associa um *nonce* à identidade de outro agente. Isto acontece pois, ao receber posteriormente uma mensagem, possivelmente uma chave de sessão, juntamente com o referido *nonce*, o agente deve ter meios de conferir com qual outro agente aquela chave deve ser usada para comunicação. Para associar um *nonce* a uma identidade, um agente deve criptografá-lo, juntamente com tal identidade, usando uma chave compartilhada com outro agente de sua confiança. Dessa forma, fica garantido que a identidade do segundo agente não será substituída na transmissão. Ao receber posteriormente uma mensagem, possivelmente a chave de sessão, juntamente com o *nonce*, ambos criptografados com uma chave compartilhada com um outro agente confiável, o primeiro agente pode acreditar que tal chave é adequada para comunicação com o agente cujo *nonce* foi relacionado anteriormente. Um exemplo dessa técnica é encontrado no protocolo Heintze-Tygar (HEINTZE; TYGAR, 1992), cujo primeiro e último passos são escritos em notação padrão como:

$$
\begin{aligned}
&1.\ A \rightarrow B : A, B, N_a, \{N_a, A, B\}K_{as} \\
&\vdots \\
&4.\ B \rightarrow A : \{N_a, K_{ab}\}K_{as}
\end{aligned}
$$

Assumindo que o agente $\boldsymbol{s}$ é confiável, o agente A, ao receber a mensagem $\{N_a, K_{ab}\}K_{as}$, tem garantias que a chave de sessão K_{ab} foi gerada para conversação com B, já que essa identidade foi associada ao *nonce* N_a no primeiro passo. Como dito anteriormente, para nós não basta assumir implicitamente que a associação é feita. É preciso que o autor especifique explicitamente tal ação, sob pena de obter uma falha de correspondência (MARRERO; CLARKE; JHA, 1997) seguindo a abordagem aqui proposta. Para o caso acima, escrevemos as seguintes sentenças:

$$
\begin{aligned}
&E.I\ A \rightarrow B : A.B.N_A^E.\{N_A^E.A.B\}K_{As}, \\
&E.I\ A \underline{\overset{N_A^E}{\sim}} B, \\
&\vdots \\
&E.I\ B \rightarrow A : \{N_a^E, K_{ab}^E\}K_{as} \Leftarrow \ldots \\
&\vdots \\
&E.I\ A \underline{\overset{K_{ab}^E}{\leftrightarrow}} B \Leftarrow \ldots
\end{aligned}
$$

a relação $\sim$ é usada para indicar a associação do *nonce* N_A^E com a identidade B. A relação $\leftrightarrow$ é usada para indicar que o agente A considera a chave K_{ab}^E adequada para comunicação com B. Se o autor não especificar explicitamente essa última ação, a análise vai novamente acusar um erro de correspondência (ver 5.4.1).

Uma outra forma de saber a identidade do agente para o qual uma chave de sessão é adequada para comunicação acontece quando o terceiro confiável criptografa, junto com a chave de sessão, as identidades dos agentes aos quais a chave se destina. Um exemplo disso está presente nos primeiros passos do protocolo Needham-Schroeder:

$$1.\ A \rightarrow S : A, B, N_a$$
$$2.\ S \rightarrow A : \{N_a, B, K_{ab}, \{K_{ab}, A\}K_{bs}\}K_{as}$$

Aqui, espera-se que A receba uma mensagem criptografada com uma chave que ele compartilha com um agente confiável (no caso S). Como a identidade de B vem criptografada junto com a chave de sessão K_{ab}, A pode deduzir que tal chave foi gerada por S para comunicação com B, e não com outro agente qualquer.

4.7 Fórmulas que Representam Falhas

Relembrando a estrutura geral de detecção de falhas, estamos interessados em demonstrar que certas fórmulas, as quais representam falhas, são conseqüências lógicas do conjunto de fórmulas da especificação do protocolo, unido com o conjunto de axiomas proposto. Resta então definir quais as fórmulas que representam falhas. As falhas estão diretamente relacionadas com os objetivos de um protocolo. Como não incluímos a representação dos objetivos na $\mathcal{L}_{ep}$ (ver 4.2), é impossível caracterizar uma falha considerando-se apenas a sentença que supostamente a representa, ou seja, uma sentença pode caracterizar uma falha num contexto mas não em outro, dependendo dos objetivos propostos. Sendo assim, deixamos por conta do analista a tarefa de elaborar quais sentenças vão representar possíveis falhas para um determinado protocolo. A abordagem proposta provê o mecanismo formal que permite a demonstração que tais sentenças são (ou não) conseqüência lógica da especificação do protocolo.

Vamos propor uma extensão à linguagem $\mathcal{L}_{ep}$ na qual criamos uma classe de sentenças para representar falhas. Essa classe será formada por um subconjunto das fórmulas da $\mathcal{L}_{ep}$, sendo que o quantificador universal não está implicitamente presente. Em seu lugar vamos escrever explicitamente o quantificador existencial ($\exists$). Para facilitar a definição das fórmulas que representam falhas, vamos recorrer à definição de fórmulas atômicas apresentada na seção 4.3. Formalmente essa nova classe de fórmulas pode ser descrita como:

Definição 4.7. *Se φ é uma fórmula atômica e $v_1, v_2, \cdots, v_n$ são as variáveis que aparecem em φ, então $\exists\ v_1, v_2, \cdots, v_n\ \varphi$ é uma fórmula para descrever falhas.*

Assim como outras sentenças da $\mathcal{L}_{ep}$, fórmulas que representam falhas podem ser escritas em mais de uma linha (ver 4.3) da seguinte forma:

$$\exists\, v_1, v_2, \cdots, v_n$$
$$\varphi$$

As fórmulas que representam falhas não devem ser usadas na especificação de protocolos. Essa restrição advém da própria estrutura dessas fórmulas, as quais, por fazerem uso do quantificador existencial, podem causar erros de subespecificação, ou seja, limitar as possibilidades de busca por uma falha. Essas sentenças podem ser vistas como **esquemas de falhas**, ou seja, se substituirmos as variáveis por constantes, podemos obter diferentes **instâncias de fórmulas**, cada uma delas representando um cenário de ataque diferente.

Para ilustrar a tarefa de identificar sentenças que representam falhas, vamos considerar os protocolos de autenticação e distribuição de chaves, os quais têm objetivos bem conhecidos (ver 2.6). Vamos usar uma classificação geral para as falhas, em vez de uma mais específica como a proposta por Carlsen (1994a). Trataremos falhas de confidencialidade, autenticidade e integridade. Apresentaremos algumas sentenças que representam falhas de cada uma dessas classes. Essa lista de **modelos** de falhas não pretende ser exaustiva, logo podem existir outras sentenças que representam falhas dessas classes e que podem ser descritas de outra forma que não a apresentada.

4.7.1 Falhas de Confidencialidade

Falhas deste tipo acontecem quando um agente A acredita que uma mensagem M é boa para servir como chave de sessão com um agente B, sendo que um outro agente C possui tal mensagem. Formalmente escrevemos:

$$\exists\, E_1, I_1, I_2, A, B, C, M$$
$$E_1.I_1\, A \overset{M}{\leftrightarrow} B,$$
$$E_1.I_2\, C \lhd M,$$
$$C \neq A,$$
$$C \neq B$$

4.7.2 Falha de Autenticidade

Neste caso, um agente A acredita que uma mensagem K_{AC} é boa para comunicação com o agente B, sendo que a mensagem K_{AC} não foi gerada para comunicação entre A e

B, ou seja, $C \neq B$. Pode acontecer de o agente B nem mesmo ter participado do cenário de ataque. A sentença que exprime tal situação é:

$$\begin{aligned}&\exists\, E, I, A, B, C\\ &E.I\ A \overset{K^E_{AC}}{\leftrightarrow} B,\\ &C \neq B\end{aligned}$$

4.7.3 Falhas de Integridade

Se um agente A vem a acreditar que uma mensagem M é uma boa chave de sessão com B, e a mensagem M for outra qualquer que não a mensagem gerada pelo terceiro confiável para comunicação entre A e B, dizemos então que há uma falha de integridade:

$$\begin{aligned}&\exists\, E, I, A, B, M\\ &E.I\ A \overset{M}{\leftrightarrow} B,\\ &M \neq K^E_{AB}\end{aligned}$$

Existe ainda a possibilidade de combinação entre esses tipos de falhas. Se deduzirmos, por exemplo, a fórmula:

$$\begin{aligned}&\exists\, E_1, I_1, I_2, A, B, C, M\\ &E_1.I_1\ A \overset{M}{\leftrightarrow} B,\\ &E_1.I_2\ C \lhd M,\\ &C \neq A,\\ &C \neq B,\\ &M \neq K^{E_1}_{AB}\end{aligned}$$

podemos afirmar que houve uma falha de confidencialidade e de integridade ao mesmo tempo.

Como último exemplo do capítulo 5, analisamos um protocolo simples para transmissão de informações com confidencialidade de um agente para outro. Esse protocolo contém uma falha de confidencialidade que pode ser representada como uma sentença da $\mathcal{L}_{ep}$, mas que não se encaixa no modelo de falha de confidencialidade descrito acima.

4.8 Conclusões

Após a discussão sobre os tipos de falhas encontrados nos protocolos de criptografia (ver 2.6) e a revisão dos métodos de especificação e análise mais conhecidos do capítulo 3, elaboramos os princípios de projeto de uma linguagem de primeira ordem para espe-

cificação e análise (ver 4.2). Como principais requisitos que devem ser atendidos por qualquer método de especificação que se proponha a servir de base para uma análise formal, podemos citar:

- Capacidade de especificar ações e estado de conhecimento, total ou parcial (ver 4.2), dos agentes;
- Capacidade de especificar a relação de tempo entre ações, execuções e instantes;
- Capacidade de representar um ambiente hostil de comunicação, e as capacidades dos agentes malignos, tais como a possibilidade de substituição de uma mensagem por outra, inclusive de tipos diferentes;
- Capacidade de representar a dependência entre ações e entre valores de mensagens recebidas e enviadas, modelando assim o comportamento mecânico dos agentes.

Com esses requisitos em mente, criamos uma teoria de primeira ordem para especificar e analisar protocolos de criptografia. As sentenças da linguagem proposta ($\mathcal{L}_{ep}$) representam ações e estado de conhecimento parcial (posse de mensagens) dos agentes. Todas as ações são parametrizadas com o instante em que ocorrem numa execução do protocolo, assim, é possível representar uma relação de ordem temporal entre as ações através do conectivo da implicação. Os agentes e mensagens são representados por variáveis quantificadas universalmente, exprimindo a possibilidade de substituição entre eles. A dependência entre valores de mensagens recebidas e enviadas é representada também com o uso de variáveis e do conectivo da implicação. Usamos uma linguagem poli-sortida para impedir, até certo ponto, a substituição entre mensagens que acarreta em sentenças sem significado, tais como *nonces* enviando instantes. Contudo, as restrições estruturais da $\mathcal{L}_{ep}$ foram suficientemente permissivas a ponto de representar as possibilidades de substituição que podem resultar em falhas. O conjunto de axiomas proposto (ver 4.5) representa o ambiente de comunicação que envolve os agentes e mensagens. Aspectos da realidade, tanto positivos (por exemplo a transmissão normal) como negativos (personificação), os quais julgamos importantes para verificação de falhas, são modelados por axiomas distintos. A combinação desses axiomas, aliada a um mecanismo de dedução (cálculo de seqüentes), leva a possibilidade de checagem formal de falhas. A metodologia de construção desse conjunto de axiomas mostrou-se bastante flexível, permitindo a inclusão de novos axiomas e até alterações nos existentes sem maiores prejuízos. Na verdade, boa parte dos axiomas propostos foram frutos desse processo de aprendizagem, e tiveram sua necessidade

demonstrada somente após nos depararmos com a impossibilidade de prosseguirmos em determinadas demonstrações quando da aplicação do método.

Discutimos ainda como a $\mathcal{L}_{ep}$ pode ser usada para especificar protocolos e as sentenças que representam possíveis falhas.

No capítulo seguinte, aplicaremos a abordagem proposta na análise de protocolos clássicos e com falhas bem conhecidas.

5 ANALISANDO PROTOCOLOS

Para ilustrar as capacidades da $\mathcal{L}_{ep}$ e da abordagem proposta no capítulo anterior, apresentaremos demonstrações do processo de especificação e análise de alguns protocolos clássicos, os quais contêm falhas bem conhecidas.

Inicialmente, analisamos um protocolo simples, proposto em Nesset (1990) especialmente para comprovar uma deficiência da lógica BAN (BURROWS; ABADI; NEEDHAM, 1990) como ferramenta de análise. Além de demonstrar que o método proposto não sofre de tal deficiência, esse exemplo ressalta a dificuldade em apresentar provas longas no formato de árvores de seqüentes. Essa dificuldade foi contornada usando-se o algoritmo proposto na seção B.1 para reescrever árvores de prova. A seguir, examinamos o protocolo Needham-Schroeder (NEEDHAM; SCHROEDER, 1978), o qual contém uma falha de frescor bem conhecida (ver 2.6). Tal protocolo originou a discussão sobre a aplicação de métodos formais para análise de protocolos de criptografia (ver 3.1). Avaliamos uma especificação proposta por nós mesmos para esse protocolo e sugerimos modificações. Demonstramos que algumas modificações causam novas falhas e outras são indiferentes.

O próximo protocolo analisado é o Otway-Rees, descrito em (SCHNEIER, 1996, p. 59). Esse é um protocolo interessante pois, em sua forma original, contém duas falhas de classes diferentes. Mais uma vez, modificações na especificação são propostas e avaliadas.

O último protocolo analisado, o protocolo de três passos, pertence a uma classe diferente daquela dos dois anteriores. Por não se tratar de um protocolo de autenticação e distribuição de chaves, sua análise ilustra a adequação do método proposto à demonstração de falhas no atendimento do requisito de confidencialidade.

Por fim, uma comparação entre os resultados aqui obtidos e os de outras abordagens é apresentada.

5.1 Revisão do Método

De acordo com a visão geral apresentada na seção 4.1 e com as definições da $\mathcal{L}_{ep}$ (ver 4.6), Δ_{ep} (ver 4.5) e de fórmulas para especificar falhas (ver 4.7), vamos relembrar a estrutura da abordagem proposta para verificação de falhas em protocolos de criptografia.

Definição 5.1. *Seja φ uma fórmula para especificar falhas. Seja Φ uma especificação de um protocolo escrita em $\mathcal{L}_{ep}$. A abordagem proposta consiste em verificar se φ é teorema da teoria deduzível a partir do conjunto de axiomas Δ_{ep} unido com a especificação Φ, ou seja, estamos interessados em responder a pergunta:*

$$\Delta_{ep} \cup \Phi \vdash \varphi \ ?$$

Na verdade, não estamos interessados somente numa resposta **sim** ou **não** para a dúvida se um protocolo apresenta uma determinada falha. No caso da falha existir, sua demonstração pode ser entendida como uma descrição da execução, ou execuções (no caso de existir mais de uma prova do teorema em questão), onde tal falha ocorre. Essa descrição constitui o que chamamos **cenário de ataque**.

Lembramos que estaremos usando o cálculo de seqüentes (SZABO, 1969) como método de prova de teoremas.

5.2 Protocolo de Nesset

Para ilustrar o uso (e a necessidade) do algoritmo proposto na seção B.1, vamos analisar o protocolo proposto por Nesset (1990), descrito em notação padrão em Carlsen (1994b, p. 19) como:

$$1.\ A \rightarrow B : \{N_a, K_{ab}\}K_a^{-1}$$
$$2.\ B \rightarrow A : \{N_b\}K_{ab}$$

O suposto objetivo básico desse protocolo é a distribuição de uma chave de sessão gerada pelo agente A para o agente B, sem o uso de um terceiro agente. A chave de sessão é assinada com a chave privada de A antes da transmissão para B. Obviamente, tal protocolo contém uma falha elementar de confidencialidade, já que essa não é uma das propriedades de uma mensagem assinada com uma chave privada (ver 2.3.5). Qualquer agente pode usar a chave pública do iniciador para descriptografar a mensagem

$\{N_a, K_{ab}\}K_a^{-1}$ e obter a chave de sessão K_{ab}. Carlsen (1994b, p. 19) afirma ainda que, na verdade, esse protocolo não foi criado com a real intenção de distribuir uma chave de sessão com confidencialidade, e sim para demonstrar uma das limitações da lógica BAN como ferramenta de análise.

Propomos a seguinte especificação em $\mathcal{L}_{ep}$ para esse protocolo:

$$
\begin{array}{ll}
\text{P1} & E.I\, A \rightarrow B : [N_A^E.K_{AB}^E]K_A^{-1} \\
\text{P2} & E.I\, B \rightarrow A : \{N_B^E\}CH \Leftarrow \\
 & \quad E.I_2\, B \leftarrow A : [NC.CH]K_A^{-1}, \\
 & \quad I_2 < I
\end{array}
$$

P1 e P2 são rótulos para futuras referências às fórmulas. Chamaremos tal especificação de Φ_n, ou seja:

$$\Phi_n = \{\text{P1, P2}\}$$

O protocolo se inicia quando um agente qualquer (A) envia para outro agente (B) um *nonce* e uma chave de sessão para comunicação futura, ambos criados por ele próprio e assinados com sua chave privada. Esse passo é o representado na sentença P1. Em P2, representamos o fato do agente respondedor (B) dar prosseguimento ao protocolo enviando sua resposta ao iniciador (A), resposta essa formada por um *nonce* criado por ele e criptografado com a chave de sessão recebida. Ainda em P2, percebe-se que a continuidade do protocolo está condicionada ao recebimento de uma mensagem assinada, e que o agente que assinou a mensagem deve ser o mesmo de quem o respondedor recebeu a mensagem (garantia essa vinda do uso da mesma variável para representar agentes (A) na sentença. Percebe-se que os *nonces* trocados pelos agentes não exercem sua função tradicional de garantir o frescor das mensagens, vez que não é feita nenhuma verificação de *nonces* recebidos com valores armazenados anteriormente. Entende-se que essa é uma decorrência do objetivo do protocolo proposto, que como dito antes, não é propriamente o de distribuir uma chave de sessão.

Vamos enunciar o teorema cuja prova demonstra formalmente a falha elementar citada acima.

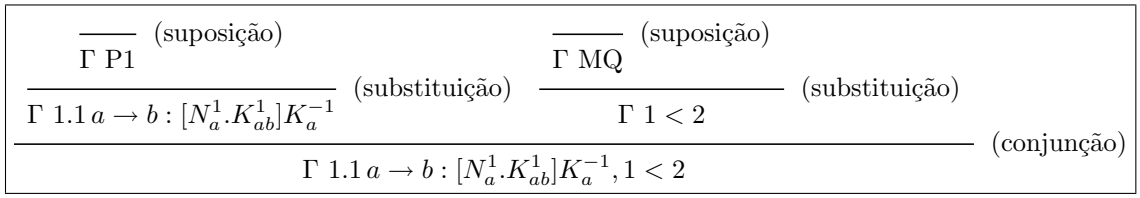

Figura 9: Demonstração do teorema 5.1, parte 1.

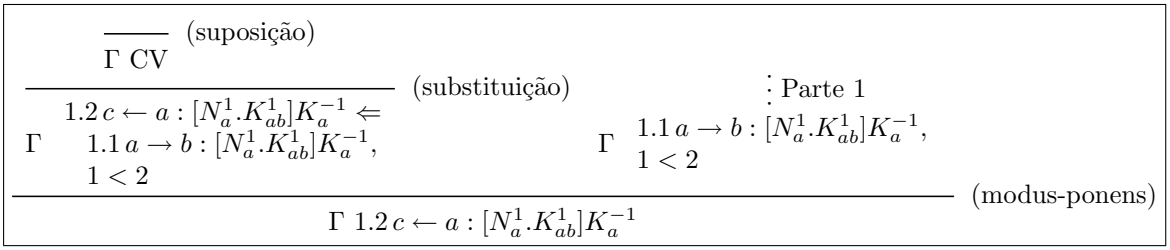

Figura 10: Demonstração do teorema 5.1, parte 2.

Teorema 5.1. *Seja φ a seguinte fórmula para representar falhas:*

$$\begin{aligned}&\exists\ E, I, C, A, B\\ &E.I\ C \lhd K^E_{AB},\\ &C \neq A,\\ &C \neq B\end{aligned}$$

Então:

$$\Phi_n \cup \Delta_{ep} \vdash \varphi$$

A fórmula φ representa o fato de que, existe um agente C, diferente do iniciador A e do respondedor B, que vem a possuir a chave de sessão K^E_{AB} entre o iniciador e o respondedor, em algum instante I dessa mesma execução E.

Demonstração. Seja Γ uma seqüência qualquer de todas as fórmulas de $\Phi_n \cup \Delta_{ep}$. Obviamente, Γ contém todas as fórmulas da especificação do protocolo e do conjunto de axiomas proposto na seção 4.5. Para qualquer fórmula ψ de $\Phi_n \cup \Delta_{ep}$, podemos sempre obter a seqüência $\Gamma\ \psi$ através da regra da suposição, conforme descrito no apêndice B. A dedução dessas seqüências vão estar nas folhas da árvore de prova de φ. Por falta de espaço nestas páginas, dividimos a árvore em 8 partes, mostradas nas figuras de 9 a 16. □

$$\dfrac{\begin{array}{cc}\vdots \text{ Parte 2} & \dfrac{\dfrac{}{\Gamma \text{ PR}}\ (\text{suposição})}{\begin{array}{c}1.2\, c \lhd [N_a^1.K_{ab}^1]K_a^{-1} \Leftarrow \\ 1.2\, c \leftarrow a : [N_a^1.K_{ab}^1]K_a^{-1}\end{array}}\ (\text{substituição}) \\ \Gamma\ 1.2\, c \leftarrow a : [N_a^1.K_{ab}^1]K_a^{-1} & \Gamma \end{array}}{\Gamma\ 1.2\, c \lhd [N_a^1.K_{ab}^1]K_a^{-1}}\ (\text{modus-ponens})$$

Figura 11: Demonstração do teorema 5.1, parte 3.

$$\dfrac{\dfrac{\dfrac{}{\Gamma \text{ PCB}}\ (\text{suposição})}{\Gamma\ 1.2\, c \lhd K_a}\ (\text{substituição}) \qquad \begin{array}{c}\vdots \text{ Parte 3} \\ \Gamma\ 1.2\, c \lhd [N_a^1.K_{ab}^1]K_a^{-1}\end{array}}{\Gamma\ 1.2\, c \lhd [N_a^1.K_{ab}^1]K_a^{-1}, 1.2\, c \lhd K_a}\ (\text{conjunção})$$

Figura 12: Demonstração do teorema 5.1, parte 4.

Fica clara a inviabilidade de apresentar provas no formato de árvores, já que, mesmo para uma prova curta, precisaríamos de uma área muito extensa para escrevê-la com clareza. Aplicando o algoritmo da seção B.1, e escolhendo, quando necessário, as hipóteses a serem aplanadas, da esquerda para a direita (vendo a árvore com a raiz para baixo), obtemos o seguinte aplanamento:

D1	Γ MS	suposição
D2	$\Gamma\ 1.2\, c \lhd N_a^1.K_{ab}^1 \Leftrightarrow$ $1.2\, c \lhd N_a^1$, $1.2\, c \lhd K_{ab}^1$	D1, substituição
D3	$\Gamma\ 1.2\, c \lhd N_a^1, 1.2\, c \lhd K_{ab}^1 \Leftarrow$ $1.2\, c \lhd N_a^1.K_{ab}^1$	D2, bi-implicação
D4	Γ MA	suposição
D5	$\Gamma\ 1.2\, c \lhd N_a^1.K_{ab}^1 \Leftarrow$ $1.2\, c \lhd [Na.K_{ab}^1]K_a^{-1}$, $1.2\, c \lhd K_a$	D4, substituição
D6	Γ PCB	suposição
D7	$\Gamma\ 1.2\, c \lhd K_a$	D6, substituição

$$\dfrac{\begin{array}{cc}\dfrac{\dfrac{}{\Gamma \text{ MA}}\ (\text{suposição})}{\begin{array}{c}1.2\, c \lhd N_a^1.K_{ab}^1 \Leftarrow \\ 1.2\, c \lhd [N_a^1.K_{ab}^1]K_a^{-1}, \\ 1.2\, c \lhd K_a\end{array}}\ (\text{substituição}) & \begin{array}{c}\vdots \text{ Parte 4} \\ \Gamma\ 1.2\, c \lhd [N_a^1.K_{ab}^1]K_a^{-1}, \\ 1.2\, c \lhd K_a\end{array} \\ \Gamma & \end{array}}{\Gamma\ 1.2\, c \lhd N_a^1.K_{ab}^1}\ (\text{modus-ponens})$$

Figura 13: Demonstração do teorema 5.1, parte 5.

D8	Γ CV	suposição
D9	Γ $1.2\, c \leftarrow a : [N_a^1.K_{ab}^1]K_a^{-1} \Leftarrow$ $1.1\, a \rightarrow b : [N_a^1.K_{ab}^1]K_a^{-1}$, $1 < 2$	D8, substituição
D10	Γ P1	suposição
D11	Γ $1.1\, a \rightarrow b : [N_a^1.K_{ab}^1]K_a^{-1}$	D10, substituição
D12	Γ MQ	suposição
D13	Γ $1 < 2$	D12, substituição
D14	Γ $1.1\, a \rightarrow b : [N_a^1.K_{ab}^1]K_a^{-1}, 1 < 2$	D11, D13, conjunção
D15	Γ $1.2\, c \leftarrow a : [N_a^1.K_{ab}^1]K_a^{-1}$	D14, D9, modus-ponens
D16	Γ PR	suposição
D17	Γ $1.2\, c \lhd [Na.K_{ab}^1]K_a^{-1} \Leftarrow$ $1.2\, c \leftarrow a : [N_a^1.K_{ab}^1]K_a^{-1}$	D16, substituição
D18	Γ $1.2\, c \lhd [Na.K_{ab}^1]K_a^{-1}$	D15, D17, modus-ponens
D19	Γ $1.2\, c \lhd [Na.K_{ab}^1]K_a^{-1}, 1.2\, c \lhd K_a$	D18, D7, conjunção
D20	Γ $1.2\, c \lhd N_a^1.K_{ab}^1$	D19, D5, modus-ponens
D21	Γ $1.2\, c \lhd N_a^1, 1.2\, c \lhd K_{ab}^1$	D20, D3, modus-ponens
D22	Γ $1.2\, c \lhd K_{ab}^1$	D21, conjunção
D23	Γ IU	suposição
D24	Γ $c \neq a$	D23, substituição
D25	Γ $c \neq b$	D23, substituição
D26	Γ $c \neq a, c \neq b$	D24, D25, conjunção
D27	Γ $1.2\, c \lhd K_{ab}^1, c \neq a, c \neq b$	D22, D26, conjunção
D28	Γ $\exists\, E, I, C, A, B$ $E.I\, C \lhd K_{AB}^E, C \neq A, C \neq B$	D27, existencial

Como explicado no apêndice B, cada seqüente do aplanamento é precedido de um rótulo, formado pela letra **D** imediatamente seguida de um número natural seqüencial (para diferenciar os rótulos), a ser utilizado nas próximas referências. No fim de cada linha, estão as regras do cálculo de sequëntes utilizadas na dedução.

Essa não é a única possibilidade de aplanamento, sendo as demais apenas variações sintáticas dessa, visto que a ordem das linhas pode mudar, de acordo com a ordem na qual as hipóteses forem aplanadas (ver B.1).

O cenário da falha formalmente descrita acima pode ser descrito como se segue.

Em D1, recorremos ao axioma MS para que logo em seguida, em D2 e D3, possamos registrar o fato de que, ao possuir a mensagem $N_a^1.K_{ab}^1$, o agente maligno $\boldsymbol{c}$ também pos-

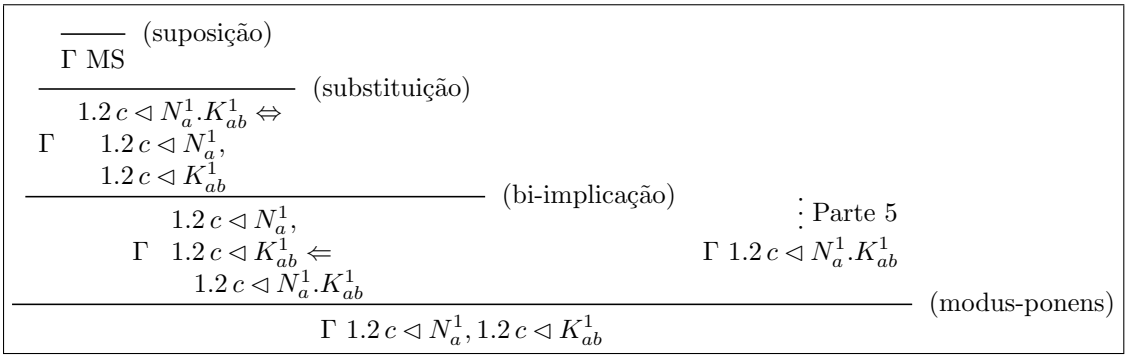

Figura 14: Demonstração do teorema 5.1, parte 6.

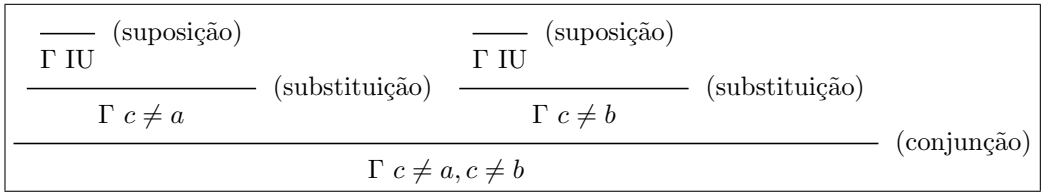

Figura 15: Demonstração do teorema 5.1, parte 7.

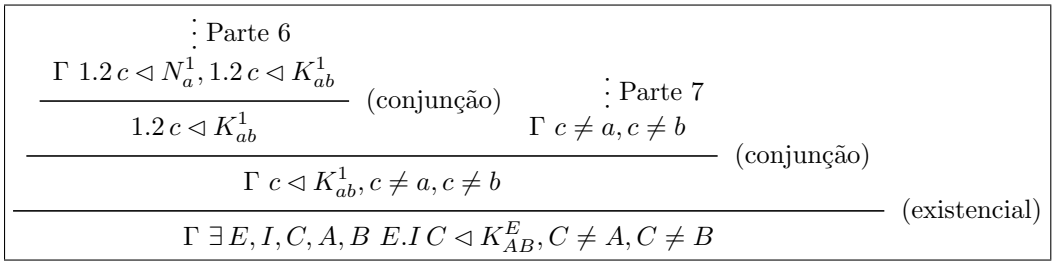

Figura 16: Demonstração do teorema 5.1, parte 8.

suirá o *nonce* N_a^1 e a chave K_{ab}^1. Em D5, o axioma MA, inserido na dedução em D4, é usado para chegarmos a constatação, já esperada, de que, ao possuir a mensagem assinada $[N_a^1.K_{ab}^1]K_a^{-1}$, o agente maligno $\boldsymbol{c}$ terá acesso a mensagem $N_a^1.K_{ab}^1$, uma vez que a confidencialidade não é uma das propriedades de uma mensagem assinada. D6 e D7 demonstram que o agente maligno $\boldsymbol{c}$, assim como qualquer outro agente, é possuidor da chave pública do agente $\boldsymbol{a}$. D9 recorre a D8 para indicar que, num momento posterior ao envio por parte do agente $\boldsymbol{a}$ da mensagem $[N_a^1.K_{ab}^1]K_a^{-1}$ para $\boldsymbol{b}$, é razoável deduzir que o agente maligno $\boldsymbol{c}$ terá acesso à mensagem enviada, o que retrata a natureza promíscua dos canais de comunicação num ambiente de comunicação hostil. As deduções até aqui são apenas preparatórias para às que se seguem. Em D10 e D11, damos início à dedução propriamente dita do cenário de ataque proposto, indicando o envio da mensagem $[N_a^1.K_{ab}^1]K_a^{-1}$, criada pelo agente $\boldsymbol{a}$, para o agente $\boldsymbol{b}$. D12, D13 e D14 servem apenas para preparar a formalização posterior de que o recebimento se dá, obviamente, após o envio. São passos inócuos, porém imprescindíveis considerando-se um sistema formal como o cálculo de seqüentes. Outras deduções desse tipo estarão presentes nas demais demonstrações formais de deste trabalho. D15 registra um passo importante. Aqui, o agente maligno $\boldsymbol{c}$ recebe a mensagem $[N_a^1.K_{ab}^1]K_a^{-1}$, vez que essa foi enviada pelo agente $\boldsymbol{a}$. A passagem da fase de recebimento para a posse da referida mensagem é formalizada em D16, D17 e D18. Chegamos ao ponto em que $\boldsymbol{c}$ possui a mensagem assinada enviada por $\boldsymbol{a}$. Como $\boldsymbol{c}$ já possui a chave pública de $\boldsymbol{a}$ (D7) e a mensagem assinada (D18), é direta a dedução da conjunção apresentada em D19. Na verdade, esse é também uma passo preparatório para a dedução seguinte, D20, a qual representa a posse por parte de $\boldsymbol{c}$ do *nonce* concatenado com a chave de sessão enviados por $\boldsymbol{a}$ no início da execução.

As deduções seguintes são também formalizações burocráticas, porém necessárias, do fato de que o agente maligno $\boldsymbol{c}$ passa a ser possuidor da chave de sessão criada por $\boldsymbol{a}$ e distribuída para $\boldsymbol{b}$, bem como do fato de tratar-se de um agente diferente de $\boldsymbol{a}$ e de $\boldsymbol{b}$.

5.3 Protocolo Needham-Schroeder

O exemplo clássico de uma falha de frescor é encontrado no protocolo Needham-Schroeder, cuja especificação em notação padrão e uma descrição em linguagem natural de tal falha podem ser encontrados no capítulo 2. Uma especificação possível em $\mathcal{L}_{ep}$ é:

SI1 $E.I\, a \lhd K_{as}$

SI2 $E.I\, s \lhd K_{as}$

SI3 $E.I\, b \lhd K_{bs}$

SI4 $E.I\, s \lhd K_{bs}$

SI5 $E.I\, s \lhd K^E_{ab}$

P1 $E.I\, A \boxtimes N^E_A,$
$E.I\, A \overset{N^E_A}{\sim} B$

P2 $E.I\, A \rightarrow s : A.B.N^E_A \Leftarrow$
$E.I\, A \boxtimes N^E_A,$
$E.I\, A \overset{N^E_A}{\sim} B$

P3 $E.I_1\, S \rightarrow A : \{NC.AG_2.K^E_{AG_1AG_2}.\{K^E_{AG_1AG_2}.AG_1\}K_{AG_2S}\}K_{AG_1S} \Leftarrow$
$E.I_2\, S \leftarrow A : AG_1.AG_2.NC,$
$E.I_1\, S \lhd K_{AG_1S},$
$E.I_1\, S \lhd K_{AG_2S},$
$E.I_1\, S \lhd K^E_{AG_1AG_2},$
$I_2 < I_1$

P4 $E.I_1\, A \rightarrow AG : M,$
$E.I_1\, A \overset{CH_1}{\leftrightarrow} AG \Leftarrow$
$E.I_2\, A \leftarrow S : \{NC.AG.CH_1.M\}CH_2,$
$E.I_1\, A \lhd CH_2,$
$E.I_3\, A \boxtimes NC,$
$E.I_3\, A \overset{NC}{\sim} AG,$
$I_2 < I_1,$
$I_3 < I_2$

P5 $E.I_1\, B \rightarrow AG : \{N^E_B\}CH_1,$
$E.I_1\, B \boxtimes F(N^E_B),$
$E.I_1\, B \overset{F(N^E_B)}{\sim} AG \Leftarrow$
$E.I_2\, B \leftarrow A : \{CH_1.AG\}CH_2,$
$E.I_1\, B \lhd CH_2,$
$I_2 < I_1$

P6 $E.I_1\, A \rightarrow B : \{F(NC)\}CH \Leftarrow$
$E.I_2\, A \leftarrow B : \{NC\}CH,$
$E.I_1\, A \lhd CH,$
$I_2 < I_1$

P7 $E.I_1\, B \overset{CH}{\leftrightarrow} AG \Leftarrow$
$E.I_2\, B \leftarrow A : \{M\}CH,$
$E.I_1\, B \lhd CH,$
$E.I_3\, B \boxtimes M,$

$$E.I_3\, B \overset{M}{\sim} AG,$$
$$I_2 < I_1,$$
$$I_3 < I_2$$

Chamaremos tal especificação de Φ_{ns}, ou seja:

$$\Phi_{ns} = \{\text{SI1, SI2, SI3, SI4, SI5, P1, P2, P3, P4, P5, P6, P7}\}$$

A sentença P1 exprime a possibilidade de um agente qualquer armazenar e relacionar um *nonce* criado por ele mesmo com outro agente qualquer. A sentença P2 exprime a possibilidade de um agente dar início a uma execução, bastando que ele tenha armazenado e relacionado o *nonce* sendo enviado. Note que não há outras condições para a realização da ação de envio de P2, uma vez que a mensagem enviada não possui submensagens criptografadas nem mensagens que não sejam normalmente conhecidas pelos agentes. No fim da análise desse protocolo, vamos relaxar as condições feitas em P2 e verificar que isso não traz nenhum prejuízo à segurança do protocolo.

5.3.1 Deduzindo uma Execução Normal

Vamos deduzir as seqüências que representam uma sessão normal do protocolo, ou seja, quando tudo ocorre exatamente como o projetista espera. Essa dedução é importante, pois mostra que a especificação proposta é suficiente para uma execução normal.

Teorema 5.2. *Existe pelo menos uma execução* $\boldsymbol{E}$ *onde o agente* $\boldsymbol{a}$ *vem a compartilhar com o agente* $\boldsymbol{b}$ *uma chave de sessão* $\boldsymbol{K_{ab}^{E}}$ *em algum instante e o agente* $\boldsymbol{b}$*, em algum instante, acredita que a mesma chave* $\boldsymbol{K_{ab}^{E}}$ *é boa para comunicação com* $\boldsymbol{a}$*, ou seja, existe pelo menos uma execução normal. Formalmente:*

$$\begin{aligned}\Phi_{ns} \cup \Delta_{ep} \vdash \ & \exists\ E, I_1, I_2 \\ & E.I_1\, a \overset{K_{ab}^{E}}{\leftrightarrow} b, \\ & E.I_2\, b \overset{K_{ab}^{E}}{\leftrightarrow} a\end{aligned}$$

Demonstração. Seja Γ uma seqüência qualquer de todas as fórmulas de $\Phi_{ns} \cup \Delta_{ep}$. Uma versão simplificada do aplanamento da árvore de prova do teorema 5.2 é mostrada abaixo. O aplanamento completo dessa prova e das restantes deste capítulo podem ser encontrados no apêndice A.

D5	Γ 1.1 $a \rightarrow s : a.b.N_a^1$	D2, D4, modus-ponens

D11	$\Gamma\ 1.2\, s \leftarrow a : a.b.N_a^1$	D10, D7, modus-ponens
D13	$\Gamma\ 1.3\, s \lhd K_{as}$	D12, substituição
D14	Γ SI4	suposição
D15	$\Gamma\ 1.3\, s \lhd K_{bs}$	D14, substituição
D25	$\Gamma\ 1.3\, s \rightarrow a : \{N_a^1.b.K_{ab}^1.\{K_{ab}^1.a\}K_{bs}\}K_{as}$	D22, D24, modus-ponens
D29	$\Gamma\ 1.4\, a \leftarrow s : \{N_a^1.b.K_{ab}^1.\{K_{ab}^1.a\}K_{bs}\}K_{as}$	D28, D26, modus-ponens
D31	$\Gamma\ 1.5\, a \lhd K_{as}$	D30, substituição
D40	$\Gamma\ 1.5\, a \rightarrow b : \{K_{ab}^1.a\}K_{bs},\ 1.5\, a \overset{K_{ab}^1}{\leftrightarrow} b$	D37, D39, modus-ponens
D41	$\Gamma\ 1.5\, a \rightarrow b : \{K_{ab}^1.a\}K_{bs}$	D40, conjunção
D45	$\Gamma\ 1.6\, b \leftarrow a : \{K_{ab}^1.a\}K_{bs}$	D44, D42, modus-ponens
D47	$\Gamma\ 1.7\, b \lhd K_{bs}$	D46, substituição
D54	$\Gamma\ 1.7\, b \rightarrow a : \{N_b^1\}K_{ab}^1$	D53, conjunção
D58	$\Gamma\ 1.8\, a \leftarrow b : \{N_b^1\}K_{ab}^1$	D57, D55, modus-ponens
D59	Γ PR	suposição
D61	$\Gamma\ 1.4\, a \lhd \{N_a^1.b.K_{ab}^1.\{K_{ab}^1.a\}K_{bs}\}K_{as}$	D29, D60, modus-ponens
D62	Γ PIP	suposição
D70	$\Gamma\ 1.8\, a \lhd K_{as}$	D69, D67, modus-ponens
D74	$\Gamma\ 1.8\, a \lhd N_a^1.b.K_{ab}^1.\{K_{ab}^1.a\}K_{bs}$	D71, D73, modus-ponens
D83	$\Gamma\ 1.8\, a \lhd K_{ab}^1$	D82, conjunção
D89	$\Gamma\ 1.9\, a \rightarrow b : \{F(N_b^1)\}K_{ab}^1$	D86, D88, modus-ponens
D93	$\Gamma\ 1.10\, b \leftarrow a : \{F(N_b^1)\}K_{ab}^1$	D92, D90, modus-ponens
D95	$\Gamma\ 1.6\, b \lhd \{K_{ab}^1.a\}K_{bs}$	D45, D94, modus-ponens
D97	$\Gamma\ 1.6\, b \lhd K_{bs}$	D46, substituição
D99	$\Gamma\ 1.6\, b \lhd K_{ab}^1.a$	D96, D98, modus-ponens
D103	$\Gamma\ 1.6\, b \lhd K_{ab}^1$	D102, conjunção
D109	$\Gamma\ 1.7\, b \boxtimes N_b^1,$ $1.7\, b \overset{F(N_b^1)}{\sim} a$	D53, conjunção
D117	$\Gamma\ 1.11\, b \overset{K_{ab}^1}{\leftrightarrow} a$	D114, D116, modus-ponens
D118	$\Gamma\ 1.5\, a \overset{K_{ab}^1}{\leftrightarrow} b$	D40, conjunção
D119	$\Gamma\ 1.5\, a \overset{K_{ab}^1}{\leftrightarrow} b,\ 1.11\, b \overset{K_{ab}^1}{\leftrightarrow} a$	D118, D117, conjunção
D120	$\Gamma\ \exists\ E, I_1, I_2$ $E.I_1\, a \overset{K_{ab}^1}{\leftrightarrow} b,$ $E.I_2\, b \overset{K_{ab}^1}{\leftrightarrow} a$	D119, existencial

□

Nessa simplificação, apenas as deduções intermediárias que julgamos importantes para

o entendimento da prova como uma representação de um cenário de ataque são mostradas. No restante das provas deste capítulo, Γ estará definido como no caso acima e serão mostrados aplanamentos simplificados da dedução das seqüências do tipo $\Gamma\ \varphi$, onde φ é o teorema que se deseja provar.

A execução normal implícita na demonstração acima pode ser descrita como abaixo, e pode ser confrontada com as características do protocolo descritas na seção 2.6.

Em D5, o agente $\boldsymbol{a}$ envia para o o terceiro confiável $\boldsymbol{s}$ a mensagem inicial do protocolo. Em D11 a mensagem é recebida por $\boldsymbol{s}$. D13,D14 e D15 expressam as suposições iniciais sobre posse das chaves compartilhadas entre os agentes. Em D25, o terceiro confiável $\boldsymbol{s}$ responde para o iniciador $\boldsymbol{a}$ com a mensagem criptografada já contendo a chave de sessão a ser utilizada entre $\boldsymbol{a}$ e $\boldsymbol{b}$. Ao receber a mensagem (D29) e sendo possuidor da chave compartilhada com $\boldsymbol{s}$ (D31), $\boldsymbol{a}$ dá prosseguimento ao protocolo enviando para $\boldsymbol{b}$ parte da mensagem recebida de $\boldsymbol{s}$ (D40). Neste ponto, $\boldsymbol{a}$ já acredita que $\boldsymbol{K^1_{ab}}$ é uma chave boa para comunicação com $\boldsymbol{b}$ (D40), visto que ele a recebeu diretamente de $\boldsymbol{s}$. Visando garantir o frescor da chave de sessão, ao receber a mensagem enviada por $\boldsymbol{a}$ (D45), e sendo possuidor da chave compartilhada com $\boldsymbol{s}$ (D47), o agente $\boldsymbol{b}$ cria o *nonce* N^1_b, criptografa-o com a chave de sessão K^1_{ab} envia para $\boldsymbol{a}$ (D54). Ao receber a mensagem (D58), e sendo possuidor da chave de sessão K^1_{ab} (D61,D70, D74 e D83), o agente $\boldsymbol{a}$ aplica uma função de modificação ao *nonce* recebido, criptografa o resultado com a chave de sessão K^1_{ab} e envia o resultado final para $\boldsymbol{b}$ (D89). Sendo também possuidor da chave de sessão (D95, D97, D99 e D103), o agente $\boldsymbol{b}$ pode verificar se a mensagem recebida (D93) contém realmente o valor da função de modificação do *nonce* que foi armazenado e relacionado com o agente $\boldsymbol{a}$ anteriormente (D109). Neste momento, $\boldsymbol{b}$ pode acreditar que a chave de sessão é boa para comunicação com $\boldsymbol{a}$ (D117). No restante da demonstração (D118, D119 e D120), fica formalizada a existência de um instante da execução onde há a crença mútua por parte dos agentes $\boldsymbol{a}$ e $\boldsymbol{b}$ de que a chave de sessão K^1_{ab} é boa.

5.3.2 Deduzindo uma Execução com Falha

Vejamos agora como um agente maligno $\boldsymbol{c}$ pode personificar o agente $\boldsymbol{a}$. Isso acontece quando $\boldsymbol{c}$ envia para o agente $\boldsymbol{b}$ uma chave de sessão usada numa execução anterior, nesse caso a chave $\boldsymbol{K^1_{ab}}$ usada na execução normal acima (ver 2.6).

Teorema 5.3. *Existem duas execuções distintas onde o agente* $\boldsymbol{b}$ *acredita que a mesma chave utilizada numa execução anterior (*$\boldsymbol{K^1_{ab}}$*) é boa para comunicação com o agente* $\boldsymbol{a}$*, ou seja, o agente* $\boldsymbol{b}$ *é ludibriado por um agente maligno* $\boldsymbol{c}$*, que se faz passar pelo iniciador*

a *e leva* ***b*** *a reutilizar uma chave de sessão. Formalmente:*

$$\begin{aligned}\Phi_{ns} \cup \Delta_{ep} \vdash \ & \exists\ E_1, E_2, I_1, I_2 \\ & E_1.I_1\, b \overset{K_{ab}^{E_1}}{\leftrightarrow} a, \\ & E_2.I_2\, b \overset{K_{ab}^{E_1}}{\leftrightarrow} a, \\ & E_2 < E_1\end{aligned}$$

Demonstração.

D126	$\Gamma\ 1.6\, c \lhd \{K_{ab}^1.a\}K_{bs}$	D124, D125, modus-ponens
D128	$\Gamma\ 2.1\, c \lhd \{K_{ab}^1.a\}K_{bs} \Leftarrow$ $1.6\, c \lhd \{K_{ab}^1.a\}K_{bs},$ $1 < 2$	D127, substituição
D133	$\Gamma\ 2.1\, c_a \rightarrow b : \{K_{ab}^1.a\}K_{bs}$	D130, D132, modus-ponens
D136	$\Gamma\ 2.2\, b \leftarrow c_a : \{K_{ab}^1.a\}K_{bs}$	D135, D134, modus-ponens
D141	$\Gamma\ 2.3\, b \lhd K_{bs}$	D46, substituição
D144	$\Gamma\ 2.3\, b \rightarrow a : \{N_b^2\}K_{ab}^1,$ $2.3\, b \boxtimes F(N_b^2),$ $2.3\, b \overset{F(N_b^2)}{\sim} a$	D143, D140, modus-ponens
D145	$\Gamma\ 2.3\, b \rightarrow a : \{N_b^2\}K_{ab}^1$	D144, conjunção
D148	$\Gamma\ 2.4\, c \leftarrow b : \{N_b^2\}K_{ab}^1$	D147, D146, modus-ponens
D150	$\Gamma\ 2.4\, c \lhd \{N_b^2\}K_{ab}^1$	D148, D149, modus-ponens
D151	Γ TI	suposição
D152	$\Gamma\ 2.4\, c \lhd K_{ab}^1 \Leftarrow$ $1.8\, c \lhd \{N_b^1\}K_{ab}^1,$ $1 < 2$	D151, substituição
D154	$\Gamma\ 1.7\, b \rightarrow a : \{N_b^1\}K_{ab}^1, 7 < 8$	D54, D56, conjunção
D155	$\Gamma\ 1.8\, c \leftarrow b : \{N_b^1\}K_{ab}^1$	D154, D153, modus-ponens
D157	$\Gamma\ 1.8\, c \lhd \{N_b^1\}K_{ab}^1$	D155, D156, modus-ponens
D158	$\Gamma\ 1.8\, c \lhd \{N_b^1\}K_{ab}^1,\ 1 < 2$	D157, D9, conjunção
D159	$\Gamma\ 2.4\, c \lhd K_{ab}^1$	D158, D152, modus-ponens

□

Em D126, fica formalizado que o agente maligno ***c*** também vem a possuir a mensagem enviada por ***a***, uma vez que o canal de comunicação é visível a todos os agentes. Mesmo sem possuir o conteúdo da mensagem criptografada, nada impede que essa mensagem seja preservada para ser usada numa tentativa de ataque à uma execução futura do protocolo,

como visto em D128. Em D133, $\boldsymbol{c}$ passar-se por $\boldsymbol{a}$, dando início a uma nova execução do protocolo e interagindo com $\boldsymbol{b}$. Observe que a mensagem enviada por $\boldsymbol{c}$ é a que foi preservada de uma execução anterior do protocolo entre $\boldsymbol{a}$ e $\boldsymbol{b}$. Tendo recebido a mensagem personificada por $\boldsymbol{c}$ (D136), e sendo possuidor da chave compartilhada com o terceiro confiável $\boldsymbol{s}$ (D141), $\boldsymbol{b}$ segue enviando para $\boldsymbol{a}$ um novo *nonce*, criptografado com a chave de sessão da execução anterior (D144, D145). $\boldsymbol{c}$ recebe a mensagem de $\boldsymbol{b}$ (D148) e passa a possuí-la (D150). Para continuarmos a dedução do cenário, cabe formalizar o ponto crucial para o sucesso do ataque, que é a posse por parte do agente maligno $\boldsymbol{c}$ da chave de sessão de uma execução anterior. Essa dedução é possível graças ao axioma TI (D151, D152). Sabendo que $\boldsymbol{b}$ enviou na execução passada um *nonce* criptografado com a chave de sessão anterior para $\boldsymbol{a}$ (D154), é possível deduzir que $\boldsymbol{c}$ também recebeu (D155) e possui essa mensagem (D157). Visto tratarem-se de duas execuções distintas e seqüenciais (D158), e tendo o agente maligno o tempo necessário para comprometer a chave de sessão da execução anterior, chegamos finalmente a posse dessa chave por parte de $\boldsymbol{c}$ (D159). A partir daí, é possível deduzir a continuação da execução em que $\boldsymbol{c}$ passa-se por $\boldsymbol{s}$ frente a $\boldsymbol{b}$. O restante do cenário de ataque é semelhante ao descrito anteriormente para a execução normal entre $\boldsymbol{a}$ e $\boldsymbol{b}$, sendo que agora $\boldsymbol{c}$ apresenta-se para $\boldsymbol{b}$ como sendo $\boldsymbol{a}$, o que está formalizado nas deduções seguintes (ver apêndice A).

O agente $\boldsymbol{b}$ acaba considerando $\boldsymbol{K^1_{ab}}$ uma boa chave para comunicação com $\boldsymbol{a}$, mesmo sendo essa uma chave utilizada numa sessão anterior.

Note que a utilização do axioma TI é crucial para completar a subversão do protocolo, pois o agente $\boldsymbol{c}$ deve responder ao desafio do agente $\boldsymbol{b}$ (mensagem enviada em P5), e para isso é preciso ter a posse da chave de sessão da execução anterior ($\boldsymbol{K^1_{ab}}$).

Para ilustrar o fato de que estamos, de certa forma, obrigando o autor a especificar explicitamente as ações internas, vamos retirar a especificação da ação de armazenamento da mensagem N_A^E das sentenças P1 e P2. Ficamos então com:

$$\begin{array}{ll} \text{P1} & E.I\,A \overset{N_A^E}{\sim} B \\ \text{P2} & E.I\,A \rightarrow S : A.B.N_A^E \Leftarrow \\ & \quad E.I\,A \overset{N_A^E}{\sim} B \end{array}$$

Percebemos, agora, que não conseguimos mais deduzir o teorema 5.2, nem nenhum outro cuja prova represente uma execução normal. Isso acontece pois a condição feita em P4, a respeito do armazenamento anterior do *nonce* recebido NC, não é mais atendida. Se retirarmos também essa verificação de P4, ou seja, se escrevermos

$$
\begin{aligned}
\text{P4}\quad & E.I_1\, A \rightarrow AG : M, \\
& E.I_1\, A \overset{CH_1}{\leftrightarrow} AG \Leftarrow \\
& \quad E.I_2\, A \leftarrow S : \{NC.AG.CH_1.M\}CH_2, \\
& \quad E.I_1\, A \lhd CH_2, \\
& \quad E.I_3\, A \overset{NC}{\sim} AG, \\
& \quad I_2 < I_1, \\
& \quad I_3 < I_2
\end{aligned}
$$

estaremos introduzindo uma outra falha de frescor. Nesse caso, é o iniciador A que não tem garantias que a mensagem recebida do agente $\boldsymbol{s}$ possui realmente uma chave de sessão fresca. Um *replay* de uma mensagem transmitida numa sessão anterior pode subverter o protocolo.

Teorema 5.4. *Seja Φ'_{ns} a especificação do protocolo Needham-Schroeder modificada como descrito acima. O iniciador $\boldsymbol{a}$ pode ser levado a reutilizar uma chave de sessão $\boldsymbol{K^1_{ab}}$, possivelmente comprometida, para comunicação segura com $\boldsymbol{b}$. Formalmente:*

$$
\begin{aligned}
\Phi'_{ns} \cup \Delta_{ep} \vdash \quad & \exists\ E_1, E_2, I_1, I_2 \\
& E_1.I_1\, a \overset{K^{E_1}_{ab}}{\leftrightarrow} b, \\
& E_2.I_2\, a \overset{K^{E_1}_{ab}}{\leftrightarrow} b, \\
& E_1 < E_2
\end{aligned}
$$

Demonstração. Vamos assumir que existe uma execução normal de Φ'_{ns}. Tal execução é descrita por uma outra prova para o teorema 5.2, semelhante àquela encontrada no início do apêndice A, mas com as seguintes modificações:

D2	$\Gamma\ 1.1\, a \overset{N^1_a}{\sim} b$	D1, substituição
D4	$\Gamma\ 1.1\, a \rightarrow s : a.b.N^1_a \Leftarrow$ $\quad 1.1\, a \overset{N^1_a}{\sim} b$	D3, substituição
D35	$\Gamma\ 1.4\, a \leftarrow s : \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as}$, $1.5\, a \lhd K_{as}$, $1.1\, a \overset{N^1_a}{\sim} b$	D32, D2, conjunção
D36	$\Gamma\ 1.4\, a \leftarrow s : \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as}$, $1.5\, a \lhd K_{as}$, $1.1\, a \overset{N^1_a}{\sim} b$, $4 < 5$	D35, D33, conjunção
D37	$\Gamma\ 1.4\, a \leftarrow s : \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as}$,	D36, D34, conjunção

$1.5\, a \lhd K_{as}$,
$1.1\, a \overset{N_a^1}{\sim} b$,
$4 < 5$,
$1 < 4$

D39 Γ $1.5\, a \rightarrow b : \{K_{ab}^1.a\}K_{bs}, 1.5\, a \overset{K_{ab}^1}{\leftrightarrow} b \Leftarrow$ D38, substituição
$1.4\, a \leftarrow s : \{N_a^1.b.K_{ab}^1.\{K_{ab}^1.a\}K_{bs}\}K_{as}$
$1.5\, a \lhd K_{as}$,
$1.1\, a \overset{N_a^1}{\sim} b$,
$4 < 5$,
$1 < 4$

A partir dessa execução normal, prosseguimos com o cenário de ataque:

D123 Γ $2.1\, a \rightarrow b : a.b.N_a^2$ D121, D122, modus-ponens
D126 Γ $1.4\, c \leftarrow s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$ D28, D125, modus-ponens
D133 Γ $2.2\, c \lhd \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$ D132, D131, modus-ponens
D136 Γ $2.2\, c_s \rightarrow a : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$ D133, D135, modus-ponens
D142 Γ $2.3\, a \leftarrow s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$ D139, D141, modus-ponens
D149 Γ $2.3\, a \leftarrow s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$, D148, D145, conjunção
$2.4\, a \lhd K_{as}$,
$2.1\, a \overset{N_a^2}{\sim} b$,
$3 < 4$,
$1 < 3$
D150 Γ $2.4\, a \rightarrow b : \{K_{ab}^1.a\}K_{bs}$, D149, D143, modus-ponens
$2.4\, a \overset{K_{ab}^1}{\leftrightarrow} b$
D151 Γ $2.4\, a \overset{K_{ab}^1}{\leftrightarrow} b$ D150, conjunção

□

Neste ponto, $\boldsymbol{a}$ considera a chave de sessão antiga $\boldsymbol{K_{ab}^1}$, e possivelmente comprometida, como adequada para falar com $\boldsymbol{b}$. Para completar a subversão total do protocolo, o agente maligno $\boldsymbol{c}$ deve enviar uma última mensagem para $\boldsymbol{a}$, fazendo com que esse acredite que a execução chegou ao fim:

D164 Γ $2.5\, c_b \rightarrow a : \{N_b^1\}K_{ab}^1$ D162, D163, modus-ponens
D179 Γ $2.3\, a \lhd N_a^1.b.K_{ab}^1, 2.3\, a \lhd \{K_{ab}^1.a\}K_{bs}$ D176, D178, modus-ponens
D188 Γ $2.6\, a \lhd K_{ab}^1$ D187, D185, modus-ponens
D191 Γ $2.7\, a \rightarrow b : \{F(N_b^1)\}K_{ab}^1$ D190, D170, modus-ponens

Chegamos a conclusão final de que a presença das duas sentenças relativas ao armazenamento do nonce criado por A em P1 e P4 são fundamentais para garantir os requisitos de frescor das chaves de sessão. O mesmo não pode ser dito quanto a presença das sentenças $E.I\,A \boxtimes N_A^E$ e $E.I\,A \overset{N_A^E}{\sim} B$ em P2. O condicionamento da ação de enviar com a de armazenar e relacionar, expresso na sentença P2, parece não ter influência na segurança do protocolo, apesar de que, do ponto de vista do implementador, essa parece ser a especificação plausível. Sendo assim, não vemos nenhum prejuízo para a segurança do protocolo em eliminar a sentença P2 e reescrever a sentença P1 como:

P1 $E.I\,A \boxtimes N_A^E$,
$E.I\,A \overset{N_A^E}{\sim} B$,
$E.I\,A \rightarrow s : A.B.N_A^E$

5.4 Protocolo Otway-Rees

Este protocolo é interessante, pois possui duas falhas de classes diferentes, uma de tipos e outra de ações internas (ver 2.6). Nossa proposta de especificação em $\mathcal{L}_{ep}$ para esse protocolo é:

SI1 $E.I\,a \lhd K_{as}$
SI2 $E.I\,s \lhd K_{as}$
SI3 $E.I\,b \lhd K_{bs}$
SI4 $E.I\,s \lhd K_{bs}$
SI5 $E.I\,s \lhd K_{ab}^E$
SI6 $E.I\,c \lhd K_{cs}$
SI7 $E.I\,s \lhd K_{cs}$
P1 $E.I\,A \rightarrow B : \overset{1}{N_A^E}.A.B.\{\overset{2}{N_A^E}.\overset{1}{N_A^E}.A.B\}K_{As}$,
$E.I\,A \boxtimes \overset{1}{N_A^E}$,
$E.I\,A \boxtimes \overset{2}{N_A^E}$,
$E.I\,A \overset{\overset{2}{N_A^E}}{\sim} B$
P2 $E.I\,B \rightarrow s : M_1.AG_1.AG_2.M_2.\{N_B^E.M_1.AG_1.AG_2\}K_{Bs}$,
$E.I\,B \boxtimes N_B^E$,
$E.I\,B \overset{N_B}{\sim} AG_1 \Leftarrow$
$E.I_2\,B \leftarrow A : M_1.AG_1.AG_2.M_2$,
$I_2 < I$

P3 $E.I\,S \rightarrow AG_2 : M.\{NC_1.K^E_{AG_1.AG_2}\}K_{AG_1S}.\{NC_2.K^E_{AG_1.AG_2}\}K_{AG_2S} \Leftarrow$
$E.I_2\,S \leftarrow B : M.AG_1.AG_2.\{NC_1.M.AG_1.AG_2\}CH_1.\{NC_2.M.AG_1.AG_2\}CH_2,$
$E.I\,S \lhd CH_1,$
$E.I\,S \lhd CH_2,$
$I_2 < I$

P4 $E.I\,B \rightarrow AG : M_1.M_2,\ E.I\,B \overset{CH_1}{\leftrightarrow} AG \Leftarrow$
$E.I_2\,B \leftarrow S : M_1.M_2.\{NC.CH_1\}CH_2,$
$E.I\,B \lhd CH_2,$
$E.I_3\,B \boxtimes NC,$
$E.I_3\,B \overset{NC}{\sim} AG,$
$I_2 < I,$
$I_3 < I_2$

P5 $E.I\,A \overset{CH_1}{\leftrightarrow} AG \Leftarrow$
$E.I_2\,A \leftarrow AG : M.\{NC.CH_1\}CH_2,$
$E.I\,A \lhd CH_2,$
$E.I_3\,A \boxtimes NC,$
$E.I_3\,A \boxtimes M,$
$E.I_3\,A \overset{NC}{\sim} AG,$
$I_2 < I,$
$I_3 < I_2$

Chamaremos essa especificação de Φ_{or}. Formalmente escrevemos:

$$\Phi_{or} = \{\text{SI1, SI2, SI3, SI4, SI5, SI6, SI7, P1, P2, P3, P4, P5}\}$$

5.4.1 Deduzindo uma Falha de Tipos

Teorema 5.5. *No protocolo Otway-Rees descrito acima, o iniciador (**a**) pode ser levado a usar uma mensagem CH, possuída pelo agente maligno **c** e diferente daquela criada pelo servidor de autenticação **s** (K^E_{ab}), como uma boa chave para comunicação com **b**. Formalmente:*

$$\begin{aligned}\Phi_{or} \cup \Delta_{ep} \vdash\ & \exists\ E, I_1, I_2, CH \\ & E.I_1\,a \overset{CH}{\leftrightarrow} b, \\ & CH \neq K^E_{ab}, \\ & E.I_2\,c \lhd CH\end{aligned}$$

Demonstração.

D2	$\Gamma\ 1.1\, a \rightarrow b : \overset{1}{N}{}^{1}_{a}.a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as},$	D1, substituição
	$1.1\, a \boxtimes \overset{1}{N}{}^{1}_{a},$	
	$1.1\, a \boxtimes \overset{2}{N}{}^{1}_{a},$	
	$1.1\, a \overset{\overset{2}{N}{}^{1}_{a}}{\sim} b$	
D14	$\Gamma\ 1.2\, c \leftarrow a : \overset{1}{N}{}^{1}_{a}.a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as}$	D13, D10, modus-ponens
D36	$\Gamma\ 1.2\, c \lhd \overset{1}{N}{}^{1}_{a}.a.b$	D35, D33, modus-ponens
D39	$\Gamma\ 1.2\, c_b \rightarrow a : \overset{1}{N}{}^{1}_{a}.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as}$	D31, D38, modus-ponens
D44	$\Gamma\ 1.3\, a \leftarrow c_b : \overset{1}{N}{}^{1}_{a}.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as}$	D43, D41, modus-ponens
D59	$\Gamma\ 1.3\, a \leftarrow b : \overset{1}{N}{}^{1}_{a}.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as},$	D58, D57, conjunção
	$1.4\, a \lhd K_{as},$	
	$1.1\, a \boxtimes \overset{1}{N}{}^{1}_{a},$	
	$1.1\, a \boxtimes \overset{2}{N}{}^{1}_{a},$	
	$1.1\, a \overset{\overset{2}{N}{}^{1}_{a}}{\sim} b,$	
	$3 < 4,$	
	$1 < 3$	
D60	$\Gamma\ 1.4\, a \overset{\overset{1}{N}{}^{1}_{a}.a.b}{\leftrightarrow} b$	D59, D51, modus-ponens

□

O cenário acima pode ser explicado como se segue. Após o envio da mensagem inicial por parte do iniciador $\boldsymbol{a}$ em D2, o agente maligno $\boldsymbol{c}$ recorre à interceptação para ter acesso à mensagem enviada (D14). Através da manipulação das diversas submensagens que compõem a mensagem interceptada, inclusive com posse na mensagem $\boldsymbol{\overset{1}{N}{}^{1}_{a}.a.b}$ (D36), a ser ofertada para $\boldsymbol{a}$ como chave de sessão, $\boldsymbol{c}$ personifica $\boldsymbol{b}$, que é quem deveria ser o respondedor (D39). Recebendo a mensagem personificada (D44), sendo possuidor da chave compartilhada com $\boldsymbol{s}$, tendo armazenado e relacionados os *nonces* descritos (D59), não resta mais nada para que $\boldsymbol{a}$ passe a considerar a submensagem $\boldsymbol{\overset{1}{N}{}^{1}_{a}.a.b}$ como boa para conversar com $\boldsymbol{b}$ (D60). O agente maligno pode agora conduzir toda uma sessão com $\boldsymbol{a}$ fazendo-se passar por $\boldsymbol{b}$.

5.4.2 Deduzindo Falhas de Ações Internas

Da forma como especificamos o protocolo, fica claro que a mensagem

$$M.\{NC_1.K^E_{AG_1.AG_2}\}K_{AG_1S}.\{NC_2.K^E_{AG_1.AG_2}\}K_{AG_2S}$$

especificada em P3, só é enviada caso as submensagens M da mensagem

$$M.AG_1.AG_2.\{NC_1.M.AG_1.AG_2\}CH_1.\{NC_2.M.AG_1.AG_2\}CH_2$$

tenham todas o mesmo valor, já que são representadas pela mesma variável M. O mesmo deve acontecer paras as submensagens indicadas por AG_1 e AG_2. Boyd e Mao (1993) sugerem que esse pode não ser o único entendimento possível, e a proposta original do protocolo deixa dúvidas quanto à necessidade de verificação dos valores das submensagens. Conforme mostrado na seção 2.6, se essas verificações não forem feitas, o agente $\boldsymbol{c}$ pode personificar o agente $\boldsymbol{b}$. Para formalizar essa falha, vamos reescrever a sentença P3 da seguinte forma:

$$\begin{aligned}
\text{P3}\quad & E.I\,S \rightarrow AG_2 : M_1.\{NC_1.K^E_{AG_1.AG_2}\}K_{AG_1S}.\{NC_2.K^E_{AG_1.AG_2}\}K_{AG_2S} \Leftarrow \\
& E.I_2\,S \leftarrow B : M_1.AG_1.AG_2.\{NC_1.M_2.AG_3.AG_4\}CH_1.\{NC_2.M_3.AG_5.AG_6\}CH_2, \\
& E.I\,S \lhd CH_1, \\
& E.I\,S \lhd CH_2, \\
& I_2 < I
\end{aligned}$$

Ou seja, a fórmula agora permite que as submensagens M_1, M_2 e M_3 assumam diferentes valores, uma vez que se tratam de diferentes variáveis. O mesmo acontecendo para AG_1, AG_3 e AG_5 e também para AG_2, AG_4 e AG_6. Teríamos então a seguinte falha.

Teorema 5.6. *Seja Φ'_{or} a versão do protocolo Otway-Rees modificada como descrito acima. De forma análoga a do teorema 5.5, o iniciador $\boldsymbol{a}$ pode ser enganado, e acabar acreditando que uma mensagem CH, possuída pelo agente maligno $\boldsymbol{c}$, e diferente da chave de sessão $\boldsymbol{K^E_{ab}}$ originada por $\boldsymbol{s}$, é uma boa chave para comunicação com $\boldsymbol{b}$. Formalmente:*

$$\begin{aligned}
\Phi'_{or} \cup \Delta_{ep} \vdash \quad & \exists E, I_1, I_2, CH \\
& E.I_1\, a \overset{CH}{\leftrightarrow} b, \\
& CH \neq K^E_{ab}, \\
& E.I_2\, c \lhd CH
\end{aligned}$$

Demonstração.

D2	$\Gamma\ 1.1\, a \rightarrow b : \overset{1}{N_a^1}.a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$,	D1, substituição
	$1.1\, a \boxtimes \overset{1}{N_a^1}$,	
	$1.1\, a \boxtimes \overset{2}{N_a^1}$,	
	$1.1\, a \overset{\overset{2}{N_a^1}}{\sim} b$	
D14	$\Gamma\ 1.2\, c \leftarrow a : \overset{1}{N_a^1}.a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D13, D10, modus-ponens
D55	$\Gamma\ 1.2\, c \lhd \{N_c^1.\overset{1}{N_a^1}.a.b\}K_{cs}$	D54, D53, modus-ponens
D66	$\Gamma\ 1.2\, c_a \rightarrow s : \overset{1}{N_a^1}.a.c.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}.\{N_c^1.\overset{1}{N_a^1}.a.b\}K_{cs}$	D63, D65, modus-ponens
D71	$\Gamma\ 1.3\, s \leftarrow c_a : \overset{1}{N_a^1}.a.c.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}.\{N_c^1.\overset{1}{N_a^1}.a.b\}K_{cs}$	D70, D68, modus-ponens
D85	$\Gamma\ 1.4\, s \rightarrow c : \overset{1}{N_a^1}.\{\overset{2}{N_a^1}.K_{ac}^1\}K_{as}.\{N_c^1.K_{ac}^1\}K_{cs}$	D82, D84, modus-ponens
D89	$\Gamma\ 1.5\, c \leftarrow s : \overset{1}{N_a^1}.\{\overset{2}{N_a^1}.K_{ac}^1\}K_{as}.\{N_c^1.K_{ac}^1\}K_{cs}$	D88, D86, modus-ponens
D105	$\Gamma\ 1.5\, c \lhd K_{ac}^1$	D105, conjunção
D107	$\Gamma\ 1.5\, c_b \rightarrow a : \overset{1}{N_a^1}.\{\overset{2}{N_a^1}.K_{ac}^1\}K_{as}$	D95, D106, modus-ponens
D111	$\Gamma\ 1.6\, a \leftarrow c_b : \overset{1}{N_a^1}.\{\overset{2}{N_a^1}.K_{ac}^1\}K_{as}$	D110, D108, modus-ponens
D125	$\Gamma\ 1.6\, a \leftarrow b : \overset{1}{N_a^1}\{\overset{2}{N_a^1}.K_{ac}^1\}K_{as}$,	D124, D117, conjunção
	$1.7\, a \lhd K_{as}$,	
	$1.1\, a \boxtimes \overset{2}{N_a^1}$,	
	$1.1\, a \boxtimes \overset{1}{N_a^1}$,	
	$1.1\, a \overset{\overset{2}{N_a^1}}{\sim} b$,	
	$6 < 7$,	
	$1 < 6$	
D126	$\Gamma\ 1.7\, a \overset{K_{ac}^1}{\leftrightarrow} b$	D125, D115, modus-ponens

□

Aqui, o cenário de ataque é semelhante ao anterior. Após o início de uma execução (D2), e interceptação da mensagem inicial (D14), o agente maligno $\boldsymbol{c}$ manipula a mensagem interceptada para produzir uma mensagem (D55) que será utilizada na personificação (D66). O terceiro confiável $\boldsymbol{s}$, ao receber a mensagem produzida por $\boldsymbol{c}$ (D71), e, considerando agora a não verificação das submensagens como suposto acima, em particular, a não comparação da identidade $\boldsymbol{c}$ em texto aberto com a identidade $\boldsymbol{b}$ contida nas submensagens criptografadas, dá prosseguimento ao protocolo (D85). Possuindo a chave de sessão criada por $\boldsymbol{s}$ (D105), $\boldsymbol{c}$ personifica o respondedor $\boldsymbol{b}$ (D107 e D111), levando $\boldsymbol{a}$ a acreditar que a chave $\boldsymbol{K_{ac}^1}$ é boa para comunicação com $\boldsymbol{b}$ (D125 e D126).

5.5 Protocolo de 3 Passos

Este protocolo, de acordo com Carlsen (1994b, p. 22) e Schneier (1996, p. 516), permite que dois agentes, possivelmente desonestos, conduzam um jogo de pôquer honesto através de um canal inseguro (por exemplo, uma linha telefônica). Outra aplicação proposta pelos autores é a transmissão de uma informação, de um agente para outro, através de um canal inseguro. Aqui discutiremos o protocolo com esse segundo objetivo em mente. Segue-se uma possível especificação em $\mathcal{L}_{ep}$.

SI $\quad \{\{M\}CH_1\}CH_2 \equiv \{\{M\}CH_2\}CH_1$

P1 $\quad E.I\ A \rightarrow B : \{M\}K_A^{-1}$

P2 $\quad E.I\ B \rightarrow A : \{M\}K_B^{-1} \Leftarrow$
$\quad\quad E.I_2\ B \leftarrow A : M,$
$\quad\quad I_2 < I$

P3 $\quad E.I\ A \rightarrow B : \langle M\rangle K_A^{-1} \Leftarrow$
$\quad\quad E.I_2\ A \leftarrow B : M,$
$\quad\quad I_2 < I$

Denotaremos tal especificação como Φ_{3p}, ou seja:

$$\Phi_{3p} = \{\text{SI, P1, P2, P3}\}$$

Note que não há chaves compartilhadas entre agentes, nem um agente confiável ao iniciador e ao respondedor.

Para que o protocolo funcione, é preciso que a função de criptografia simétrica tenha uma característica singular, a comutatividade entre chaves, descrita formalmente na sentença SI. Os únicos requisitos de segurança envolvidos são a confidencialidade e a integridade, ou seja, a autenticação dos agentes não é um dos objetivos do protocolo.

5.5.1 Falha de Oráculo Simples

Se considerarmos um ambiente onde cada agente só pode exercer o papel de um dos lados do protocolo (iniciador ou respondedor), podemos chegar a dedução de um cenário onde um agente maligno $\boldsymbol{c}$ acaba por conhecer a mensagem enviada originalmente para $\boldsymbol{b}$.

Teorema 5.7. *Existe uma execução do protocolo onde um agente A envia uma informação $(info)$ criptografada com sua chave privada (K_A^{-1}) para o agente B e, mesmo criptogra-*

fada, tal informação acaba sendo possuída por um agente maligno C, diferente de B. Formalmente:

$$\begin{aligned}\Phi_{3p} \cup \Delta_{ep} \vdash \quad & \exists\ E, I_1, I_2, A, B, C, M \\ & E.I_1\, A \rightarrow B : \{M\}K_A^{-1}, \\ & E.I_2\, C \lhd M, \\ & C \neq B\end{aligned}$$

Demonstração.

D2	$\Gamma\ 1.1\, a \rightarrow b : \{info\}K_a^{-1}$	D1, substituição
D8	$\Gamma\ 1.2\, c \leftarrow a : \{info\}K_a^{-1}$	D7, D4, modus-ponens
D14	$\Gamma\ 1.2\, c_b \rightarrow a : \{info\}K_a^{-1}$	D11, D13, modus-ponens
D19	$\Gamma\ 1.3\, a \leftarrow c_b : \{info\}K_a^{-1}$	D18, D16, modus-ponens
D27	$\Gamma\ 1.4\, a \rightarrow b : \langle\{info\}K_a^{-1}\rangle K_a^{-1}$	D26, D24, modus-ponens
D31	$\Gamma\ 1.5\, c \leftarrow a : \langle\{info\}K_a^{-1}\rangle K_a^{-1}$	D30, D28, modus-ponens
D33	$\Gamma\ 1.5\, c \lhd \langle\{info\}K_a^{-1}\rangle K_a^{-1}$	D31, D32, modus-ponens
D37	$\Gamma\ \langle\{info\}K_a^{-1}\rangle K_a^{-1} \equiv info$	D36, substituição
D39	$\Gamma\ 1.5\, c \lhd info$	D38, D35, modus-ponens
D41	$\Gamma\ c \neq b$	D40, substituição
D44	$\Gamma\ \exists\ E, I_1, I_2, A, B, C$ $E.I_1\, A \rightarrow B : \{info\}K_a^{-1}$, $E.I_2\, C \lhd info$, $C \neq B$	D43, existencial

□

Em D2, o iniciador ***a*** dá início a uma execução do protocolo enviando uma informação ($info$) para o respondedor ***b***. Dadas as propriedades de promiscuidade dos canais de comunicação, o agente maligno ***c*** vem a receber a mensagem enviada por ***a*** (D8). Utilizando-se da capacidade de personificação, ***c*** assume a identidade de ***b*** e envia de volta (D14) para ***a*** a mesma mensagem enviada em D2. O agente ***a***, ao receber a mensagem enviada por ***c*** (D19), responde para ***b*** (personificado pelo agente maligno) com a mensagem recebida descriptografada com sua chave privada (D27). Assim como anteriormente, ***c*** acaba recebendo a mensagem enviada (D31) e, obviamente, possuindo tal mensagem (D33). Considerando-se que ao descriptografar a mensagem recebida em D8 o agente ***a*** anulou o efeito da criptografia (D37), deduz-se que o agente maligno ***c*** acaba por possuir a informação enviada em D2 (D39). Dada a propriedade de diferença entre identidades (D41),

fica fácil deduzir o teorema desejado (D44).

Como não há nenhum tipo de autenticação, $\boldsymbol{a}$ deve estar convencido de que apenas $\boldsymbol{b}$ conhece a informação enviada, o que, como demonstrado, pode não ser o caso.

Interessante notar que esse ataque não está relacionado à propriedade de comutatividade entre chaves, expressa em SI, podendo ser aplicado mesmo se considerando uma função de criptografia simétrica sem essa característica.

5.5.2 Falha de Oráculo Múltiplo

Carlsen (1994b) afirma ainda que, se considerarmos agora um ambiente mais hostil (e mais realista), onde qualquer agente pode fazer o papel de qualquer um dos lados do protocolo, um agente maligno tem agora 4 (em vez de 2) ângulos diferentes para o ataque.

Contudo, avaliando o ataque proposto, observa-se que foi considerada, implicitamente, uma outra propriedade da a função de criptografia simétrica, que aqui chamaremos **anulação de criptografia**. Trata-se de termos o efeito da criptografia anulado quando uma mensagem é criptografada pela segunda vez com a mesma chave usada na primeira vez. Formalizaremos essa suposição propondo a adição da sentença SI2 à especificação do protocolo:

SI2 $\quad \{\{M\}CH\}CH \equiv M$

Isso posto, um desses ângulos de ataque pode ser descrito formalmente como uma nova prova para a falta de confidencialidade da informação enviada criptografada pelo iniciador.

Teorema 5.8. *Seja Φ'_{3p} a versão do protocolo de 3 passos modificada como descrito acima. Existe uma execução do protocolo onde um agente A envia uma informação ($info$) criptografada com sua chave privada (K_A^{-1}) para o agente B e, mesmo criptografada, tal informação acaba sendo possuída por um agente maligno C, diferente de B. Formalmente:*

$$\begin{aligned}\Phi'_{3p} \cup \Delta_{ep} \vdash \quad & \exists\ E, I_1, I_2, A, B, C, M \\ & E.I_1\, A \rightarrow B : \{M\}K_A^{-1}, \\ & E.I_2\, C \lhd M, \\ & C \neq B\end{aligned}$$

Demonstração.

D2	$\Gamma\ 1.1\, a \rightarrow b : \{info\}K_a^{-1}$	D1, substituição
D8	$\Gamma\ 1.2\, c \leftarrow a : \{info\}K_a^{-1}$	D7, D4, modus-ponens
D20	$\Gamma\ 1^{\|\|}.1\, c_b \rightarrow a : \{info\}K_a^{-1}$	D19, D13, modus-ponens
D25	$\Gamma\ 1^{\|\|}.2\, a \leftarrow c_b : \{info\}K_a^{-1}$	D24, D22, modus-ponens
D33	$\Gamma\ 1^{\|\|}.3\, a \rightarrow b : \{\{info\}K_a^{-1}\}K_a^{-1}$	D32, D30, modus-ponens
D37	$\Gamma\ 1^{\|\|}.4\, c \leftarrow a : \{\{info\}K_a^{-1}\}K_a^{-1}$	D36, D34, modus-ponens
D39	$\Gamma\ 1^{\|\|}.4\, c \lhd \{\{info\}K_a^{-1}\}K_a^{-1}$	D37, D38, modus-ponens
D41	$\Gamma\ \{\{info\}K_a^{-1}\}K_a^{-1} \equiv info$	D40, substituição
D45	$\Gamma\ 1^{\|\|}.4\, c \lhd info$	D44, D43, modus-ponens
D49	$\Gamma\ 1.2\, c \lhd info$	D48, D46, modus-ponens
D53	$\Gamma\ 1.1\, a \rightarrow b : \{info\}K_a^{-1}$, $1.2\, c \lhd info$, $c \neq b$	D52, D51, conjunção

□

Desta vez, após o envio da mensagem inicial por parte de ***a*** (D2) e interceptação da mensagem pelo agente maligno ***c*** (D8), inicia-se uma personificação numa execução paralela do mesmo protocolo. Neste ponto, o agente maligno faz-se passar por ***b*** no papel de iniciador da execução paralela (D20). Na execução paralela, o agente ***a*** é agora respondedor e acaba por receber a mensagem personificada (D25). Confirmando a natureza mecânica das ações dos agentes, ***a*** responde simplesmente criptografando a mensagem recebida com sua própria chave privada, sem observar que trata-se uma mensagem gerada por ele mesmo numa execução paralela do protocolo (D33). O agente maligno ***c*** intercepta essa resposta (D37 e D39) e aproveita-se da propriedade de anulação da criptografia anteriormente comentada (D41) para ter acesso a informação desejada (D45). Já que a informação foi adquirida numa execução paralela, o agente maligno pode agora retornar à execução inicial já possuidor da referida informação (D49), subvertendo completamente o objetivo da transmissão com confidencialidade.

Opcionalmente, o agente maligno ***c*** pode decidir terminar normalmente a primeira execução de forma a persuadir o agente ***a*** por completo:

D58	$\Gamma\ 1.2\, c_b \rightarrow a : lixo$	D56, D57, modus-ponens
D61	$\Gamma\ 1.3\, a \leftarrow c_b : lixo$	D60, D59, modus-ponens
D67	$\Gamma\ 1.4\, a \rightarrow b : \langle lixo \rangle K_a^{-1}$	D66, D65, modus-ponens

Em D58, o agente maligno, agindo como o agente respondedor, envia uma mensagem

qualquer, aqui representada por ***lixo***, como resposta à mensagem inicial de ***a***. Ao receber a mensagem (D61), o iniciador da primeira execução finaliza sua participação respondendo com a mensagem ***lixo*** criptografada com sua chave privada (D67).

Mais uma vez, a demonstração da falha de confidencialidade prescinde da suposição inicial SI.

5.6 Conclusões e Comparação de Resultados

Nosso objetivo principal foi comprovar a utilidade de teoria proposta, não em provar a corretude dos protocolos, e sim para demonstrar formalmente que uma determinada falha ocorre.

Vamos reescrever a tabela apresentada por Carlsen (1994b, p. 48), onde características das lógicas conhecidas são comparadas, a fim de incluir a abordagem deste trabalho, rotulada como $\mathcal{L}_{ep}$:

Característica / Lógica	BAN	GYN	KG	GS	AT	MB	$\mathcal{L}_{ep}$
Criptografia de chave secreta no escopo?	sim	sim	sim	sim	sim	sim	**sim**
Criptografia de chave pública no escopo?	sim	sim	sim	sim	não	não	**sim**
Chave compartilhada entre mais de 2 agentes?	não	não	não	não	não	não	**não**
Semântica saudável para cada operador?	não	nenhum	nenhum	nenhum	quase	nenhum	**sim**
Axiomatização finita?	não	sim	não	não	sim	sim	**não**
Encaminhamento de mensagens criptografadas?	não	sim	não	não	sim	não	**sim**
Distinção entre os componentes do conhecimento?	não	sim	não	não	sim	não	**não**
Especificação orientada a palavra?	não	não	não	não	não	não	**sim**
Analisa requisitos de confidencialidade?	não	não	não	não	alguns	alguns	**sim**

Tabela 46: Características de alguns métodos de análise.

Acreditamos que uma extensão da $\mathcal{L}_{ep}$ que incorpore chaves compartilhadas entre mais de 2 agentes não é difícil de ser conseguida.

Classificamos o sistema de axiomas proposto como infinito pois os esquemas de axiomas PE e MQ, por estarem definidos em função do conjunto dos naturais (e de $\mathcal{C}_2$), geram um número infinito de instâncias de axiomas. No capítulo seguinte, quando discutirmos aspectos de automatização de nosso método, vamos restringir o número de execuções e instantes a um valor fornecido pelo analista, que pode ser tão grande quanto se queira, ou quanto os recursos computacionais disponíveis permitam. Teremos então um número finito de instâncias dos axiomas PE e MQ, viabilizando a busca exaustiva por cenários de ataque.

Confrontando os resultados da abordagem de análise proposta com os da tabela apre-

sentada por Carlsen (1994b, p. 60), abaixo reescrita para acomodar os resultados deste trabalho, verificamos que muitos dos tipos de falhas da classificação proposta por esse mesmo autor são detectados.

Tipo de Falha / Método de Análise	NRL	G	BAN	AT	MB	CKT5	$\mathcal{L}_{ep}$
Captura falhas elementares?	sim	não	não	sim	sim	sim	**sim**
Captura falhas de dicionário?	não	sim	não	não	não	não	**não**
Especifica propriedades de frescor?	sim	não	sim	sim	sim	sim	**sim**
Captura falhas de frescor?	não	não	algumas	algumas	algumas	não	**sim**
Especifica ambientes de papel único ?	sim	não	sim	sim	sim	sim	**sim**
Captura falhas de oráculo simples?	sim	não	não	sim	não	sim	**sim**
Especifica ambientes de múltiplos papeis?	sim	não	não	não	não	sim	**sim**
Captura falhas de oráculo múltiplo?	sim	não	não	não	não	sim	**sim**
Especifica os tipos das mensagens?	sim	não	não	não	não	não	**não**
Captura falhas de tipos?	sim	não	não	não	não	algumas	**sim**
Especifica ações internas?	sim	não	não	não	não	algumas	**sim**
Capaz de analisar falhas de ações internas?	sim	não	não	não	não	não	**sim**

Tabela 47: Escopo da especificação e tipos de falhas detectados de alguns métodos de análise.

Vamos comentar cada uma de nossas afirmações positivas.

Falhas elementares são facilmente detectadas através do axioma CV, o qual representa a fragilidade dos canais de comunicação. Sempre que uma mensagem, a qual deveria ser mantida em sigilo, for transmitida sem estar criptografada, seja essa transmissão propositadamente adicionada para testar o método de análise, (como no protocolo de Nesset) ou casual, o axioma CV permite a dedução de uma sentença que representa o fato de que todos os agentes recebem tal mensagem, e conseqüentemente a possuem (fato representado pelo axioma PR). A posse de tal mensagem por um agente indevido, de acordo com os objetivos do protocolo, pode representar uma falha de confidencialidade.

A relação $\boxtimes$ foi criada especialmente para permitir a especificação das propriedades de frescor de uma mensagem. A displicência em seu uso pode acarretar num protocolo inútil ou com uma falha de frescor, como mostra a seção 5.3.2. Já seu uso correto impede a demonstração de uma falha, já que qualquer execução subversiva é barrada pela estrutura condicional das ações do protocolo.

Para especificar ambientes de papel único, o projetista deve fazer uso das constantes nas sentenças da especificação. Esta técnica fixa um determinado comportamento a um determinado agente. Se, por outro lado, for desejado permitir que um determinado papel seja realizado por qualquer agente capacitado, *i.e.*, que atenda as condições especificadas, então deve-se usar variáveis em vez de constantes. Muitas vezes, os dois casos estão presentes num mesmo protocolo, como visto na seção 5.3. A possibilidade de especificar múltiplos papéis (através do uso de variáveis), aliada a uma representação para diferenciar

a várias execuções, possivelmente simultâneas, permite-nos avaliar nosso método como capaz de demonstrar falhas de oráculo simples e múltiplo.

Como comentado abaixo, não fazemos uso de uma representação para os tipos das mensagens. O uso de variáveis na representação das mensagens, bem como a modelagem do comportamento automático dos agentes ao tratá-las, permite-nos demonstrar quando um protocolo é sensível a falhas dependentes de implementação, ou seja, quando existe a possibilidade de uma falha ocorrer, dependendo de uma codificação particular ser usada ou não. Segundo Carlsen (1994b, p. 18), esse tipo de falha pode ser considerado como dependente de implementação, já que, a partir de uma especificação de alto nível, podem surgir diferentes implementações, sendo que pelo menos uma delas contém uma falha e pelo menos uma não contém tal falha. A implementação que contém a falha é aquela que usa exatamente a codificação para as mensagens (e seus tipos) que permite um agente maligno subverter o protocolo. Destacamos a capacidade da abordagem proposta em detectar esse subconjunto de falhas dependentes de implementação. Contudo, existe uma outra classe de falhas relacionada a implementação a qual não é endereçada. Esse é o caso das falhas que podem surgir em razão do uso de uma determinada função de criptografia, a qual foi ingenuamente considerada segura (assim como é o caso em todos os métodos de análise no nível funcional), mas que contém uma fragilidade que pode ser explorada em tempo viável. Carlsen classifica essas falhas como **falhas relacionadas aos sistemas de criptografia**. Para detectar esse tipo de falha, um método de análise teria que representar o comportamento de cada sistema (ou função) de criptografia em particular, em vez de usar uma função genérica, a qual representa um sistema perfeito. A codificação usada para o formato das mensagens também deveria estar formalmente definida. Finalmente, o ponto crucial estaria em representar (no sistema de axiomas, nos casos das lógicas) as capacidades dos agentes malignos em explorar as fraquezas desses sistemas criptográficos em particular. Este último ponto é tão importante que sua concepção bastaria para detectar quando um protocolo é sensível a uma falha relacionada aos sistemas criptográficos, ou seja, quando o uso de um sistema em particular pode resultar num protocolo inseguro.

Confrontando as características de nossas especificações com os aspectos que Carlsen (1994b, p. 60–61) julga ainda imaturos em outras abordagens, teremos os seguintes comentários:

Tipos Falhas de tipos acontecem exatamente quando os tipos das mensagens não são checados. Desse ponto de vista, podemos ver as falhas de tipos como casos par-

ticulares das falhas de ações internas. Alguns protocolos mais robustos não estão sujeitos a falhas de tipos. Isso acontece quando a própria estrutura das mensagens trocadas não permite a substituição de mensagens de um tipo por outras de outro tipo. Para estes protocolos, é indiferente especificar tipos ou não, já que as falhas associadas não existem. Dado um protocolo qualquer, gostaríamos de saber se ele pertence ou não a essa classe mais robusta. A fim de detectar quando um protocolo é sensível às falhas de tipos, preferimos não especificar os tipos das mensagens nem expressar as ações internas referentes a checagem de tipos.

Ações internas Para detectar quando um protocolo é sensível às falhas de ações internas, provimos a $\mathcal{L}_{ep}$ com predicados que explicitam tais ações. A presença ou não destes predicados na especificação faz com que um dos casos abaixo ocorra:

- Existem apenas execuções normais do protocolo.
- Não existem execuções normais do protocolo.
- Existe execuções normais e execuções com falhas.

Um exemplo concreto desses casos é encontrado na análise do protocolo Otway-Rees (ver 5.4).

Suposições sobre múltiplos papéis: A capacidade de especificar ambientes de múltiplos papéis é alcançada usualmente de duas formas:

- Distinguindo-se, ou seja, especificando-se separadamente, os papeis dos agentes (iniciador, respondedor e servidor).
- Introduzindo um segundo nível na especificação, que fixa as identidades dos agentes a um conjunto de papeis possíveis.

Apesar de não termos utilizado nenhuma dessas técnica, acreditamos ser capazes de especificar e analisar questões que envolvem múltiplos papéis. Isso pois as especificações em $\mathcal{L}_{ep}$ não amarram nenhum agente em particular a um papel no protocolo, ou seja, em princípio, qualquer agente pode exercer qualquer papel. O que restringe um agente em particular a um determinado papel dentro do protocolo são as suposições iniciais feitas a respeito de tal agente e, é claro, a própria estrutura condicional das sentenças da especificação.

Orientação a mensagens Quando usamos especificações onde as sentenças não representam diretamente o conteúdo das mensagens sendo enviadas, aumentamos as possibilidades de ambigüidades na interpretação dos resultados da análise. É bem

provável também que esse tipo de especificação seja de pouca utilidade para o implementador, já que, mais uma vez, um entendimento particular, e talvez diferente do do autor, pode introduzir uma falha. As sentenças da $\mathcal{L}_{ep}$ representam diretamente o conteúdo das mensagens sendo enviadas, inclusive a ordem das submensagens é importante. Esse aspecto torna as especificações em $\mathcal{L}_{ep}$ úteis aos implementadores. É claro que existem ainda margens para definir qual a real codificação das mensagens e as funções de criptografia a serem usadas numa implementação. A escolha de uma codificação ou função em particular pode introduzir uma falha dependente de implementação.

No atual estado, a abordagem proposta não é capaz de detectar esse tipo de falha. Isto pois usamos uma interpretação para as mensagens e funções de criptografia a qual não contempla aspectos da implementação. Isso nos permite detectar falhas num nível superior, ou seja, que independem da real codificação ou função utilizada. Para analisar falhas dependentes de implementação, uma interpretação deve ser completamente definida, ou seja, o domínio das mensagens (*i.e.*, sua codificação), bem como as várias funções de criptografia, tanto simétrica como assimétricas, deveriam ser bem definidas.

Carlsen afirma que as lógicas modais de conhecimento tem se mostrado as mais adequadas para analisar requisitos de confidencialidade e integridade, enquanto que as lógicas de crenças têm sido limitadas às questões de disponibilidade (CARLSEN, 1994b, p. 61). Nesse mesmo trabalho, o autor afirma ainda que as lógicas que contêm negação possuem maior habilidade para analisar requisitos de confidencialidade e integridade (CARLSEN, 1994b, p. 51). Apesar da teoria proposta não ser modal, nem possuir negação, acreditamos ser capazes de demonstrar falhas de integridade, confidencialidade e autenticação (ou falta de correspondência). Isso é devido ao fato de que os teoremas que provamos possuem uma interpretação diretamente relacionada a esses conceitos. Quando provamos, por exemplo, um teorema da forma

$$1.7\, a \overset{K^1_{ac}}{\leftrightarrow} b,\ 1.7\, c \lhd K^1_{ac}$$

fica evidenciada a falta de confidencialidade e integridade da chave de sessão K^1_{ac}.

Uma abordagem filosoficamente semelhante a aqui descrita é o uso de *Model Checking* para análise de protocolos, como proposto em Marrero, Clarke e Jha (1997). Nesse caso, protocolos são modelados como máquinas de estado finito, e um exame de todas as

possibilidades de passagem de um estado inicial para um final pode apontar um cenário onde ocorre uma falha. Marrero, Clarke e Jha (1997) usam um modelo considerado por nós não tão intuitivo quanto o afirmado por seus autores. Nesse modelo, aqui descrito resumidamente, cada agente é representado por uma 4-tupla $< N, p, I, B >$, onde:

- N representa a identidade do agente.
- p é um programa, também chamado pelos autores de processo, (de sintaxe similar a de CSP) que representa uma seqüência de ações a serem realizadas.
- I é um conjunto de mensagens conhecido pelo agente.
- B é um mapeamento das variáveis que aparecem no programa p para as mensagens do conjunto I.

Para completar o modelo, um estado global é representado pela composição assíncrona dos processos dos agentes e de um processo de um agente maligno, mais dois contadores para marcar a diferença de vezes em que um agente A comunica-se com um agente B e, finalmente, dois conjuntos de mensagens consideradas definitivamente secretas e temporariamente secretas. Um algoritmo é então definido para, dada a representação de um protocolo, varrer todas as possibilidades de evolução dos agentes, inclusive do agente maligno, em busca de uma condição de falha. Assim como na abordagem aqui proposta, quando uma falha é encontrada, a descrição formal de uma execução subvertida pode ser apresentada. Esse método também se identifica com o deste trabalho por realizar a análise levando em consideração o ponto de vista de um agente maligno e suas capacidades. Através do usos dos contadores descritos acima, esse método parece particularmente interessado em falhas de falta de correspondência, tais como a mostrada na seção 5.4.1. As suposições iniciais parecem ter seu espaço reservado na especificação dos protocolos. Contudo, as ações internas relativas a verificação de *nonces* não encontram correspondentes nos processos dos agentes, o que nós leva a concluir a incapacidade para tratar falhas de frescor. Apesar de não demonstrado no trabalho, falhas de tipos também poderiam ser detectadas, já que existe uma representação para o mapeamento entre as variáveis dos processos e as mensagens conhecidas por um agente. Faltaria apenas a descrição da condição que caracterizaria uma falha de tipos, ou seja, quando o assinalamento de uma determinada mensagem a uma determinada variável representa uma falha. A capacidade de analisar falhas de múltiplos papéis também foi endereçada através da representação dos papéis dos agentes, a qual não prende um papel a um determidado agente. Falhas

decorrentes de execuções simultâneas também são passíveis de serem detectadas, uma vez que as ações dos processos podem ser intercaladas para formar uma execução completa.

Como qualquer outro método de análise, o deste trabalho possui um escopo finito e bem definido pela linguagem que apresentamos, ou seja, estamos nos propondo a analisar um protocolo se, e somente se, esse puder ser escrito em $\mathcal{L}_{ep}$. Não é difícil imaginar que existe toda uma gama de protocolos de criptografia que simplesmente não podem ser descritos em $\mathcal{L}_{ep}$. Não garantimos a completude do método proposto, ou seja, podem existir também protocolos que são passíveis de uma especificação em nossa linguagem e que contém falhas que simplesmente não podem ser descritas usando-se as sentenças da $\mathcal{L}_{ep}$.

O problema geral da prova de teoremas é semi-decidível, ou seja, não existe método computacional conhecido para, dada uma teoria qualquer, decidir se uma sentença é ou não teorema de tal teoria (EBBINGHAUS; FLUM; THOMAS, 1984, cap. 10). Algumas teorias em particular podem ser provadas decidíveis, isto é, um procedimento computacional que responde sim ou não a dúvida de que uma determinada sentença é ou não teorema de tal teoria, pode ser apresentado. A teoria aqui proposta não foi provada decidível. Para demonstrar tal decidibilidade, teríamos que considerar todas as possíveis especificações, já que nosso conjunto total de axiomas varia de acordo com a especificação proposta. Portanto, dada uma especificação de um protocolo qualquer e uma sentença que representa uma suposta falha, não garantimos a existência de um algoritmo, de qualquer complexidade, que decida se tal sentença é ou não provável a partir daquela especificação (e dos outros axiomas do ambiente). A garantia que podemos dar advêm dos resultados do estudo das lógicas de primeira ordem, os quais afirmam que, uma fórmula φ é conseqüência lógica de um conjunto de fórmulas Γ se, e somente se, existe uma prova de φ a partir de Γ (EBBINGHAUS; FLUM; THOMAS, 1984, cap. 5). Tal resultado nos permite concluir que, se um protocolo puder ser especificado em nossa linguagem e uma sentença representando uma falha for identificada, então nosso método é capaz de encontrar o cenário onde tal falha ocorre se, e somente se, o protocolo especificado apresenta essa falha. No capítulo seguinte, vamos usar a resolução-LSD (CASANOVA; GIORNO; FURTADO, 1987, cap. 7) para automatizar a prova de teoremas. Como não sabemos, *a priori*, se uma determinada sentença é ou não teorema da teoria formada pela especificação de um protocolo junto com os axiomas do ambiente, não podemos garantir que a tentativa de demonstrar um teorema termine em tempo finito. Sendo mais específico, se uma sentença for realmente teorema, então em algum momento a resolução-LSD termina com uma prova do teorema. Porém, se tentarmos provar que uma determinada sentença é um teorema (mas que na

verdade não é), então pode acontecer de o procedimento nunca apresentar uma resposta positiva ou negativa, isto é nunca terminar.

6 EXPERIÊNCIAS COM A AUTOMATIZAÇÃO DA ANÁLISE

Neste capítulo, descrevemos uma experiência com a implementação de uma ferramenta para dar suporte automático às provas de teoremas como mostradas no capítulo 5. A ferramenta consiste basicamente numa linguagem de programação em lógica, que por sua vez é um subconjunto da linguagem PROLOG (CLOCKSIN; MELLISH, 1987). Para que pudéssemos usar o mecanismo de prova de teoremas embutido nos interpretadores e compiladores PROLOG (WIELEMAKER, 1999), tivemos que implementar um meta-interpretador para a $\mathcal{L}_{ep}$, uma vez que as teorias formadas pelo conjunto de axiomas Δ_{ep} e as especificações de protocolos não apresentam propriedades finitárias.

Após discutirmos as problemáticas envolvidas na criação de tal sistema, mostramos como as sentenças da $\mathcal{L}_{ep}$ (ver 4.6), podem ser facilmente transformadas para cláusulas em PROLOG. Em seguida, apresentamos as cláusulas que representam alguns dos axiomas da seção 4.5. Finalmente, apresentamos um exemplo completo da especificação do protocolo de Nesset transformada em cláusulas e dos resultados de consultas ao sistema sobre a falha analisada na sessão 5.2.

6.1 Introdução

Como afirmado no início do capítulo 4, ao aliarmos a $\mathcal{L}_{ep}$ a um sistema automático de prova de teoremas, poderemos deixar por conta de tal sistema a resposta para a dúvida se um protocolo possui uma determinada falha. Adicionalmente, caso uma prova de falha seja encontrada, o sistema poderia, a partir da prova do teorema que representa tal falha, apresentar uma descrição do cenário de ataque onde tal falha ocorre.

Na tentativa de implementarmos a primeira das funcionalidades acima, decidimos criar uma **linguagem de programação em lógica com propósito específico**, que por sua vez é um subconjunto da linguagem PROLOG. Essa decisão foi baseada na possibilidade

de usarmos o mecanismo de prova de teoremas embutido nos interpretadores PROLOG, facilitando assim a criação de um protótipo em pouco tempo. O leitor mais experiente deve ter percebido que, na verdade, a $\mathcal{L}_{ep}$ foi projetada e concebida com essa possibilidade em mente.

O primeiro obstáculo a ser vencido diz respeito às propriedades não-finitárias das teorias formadas, uma vez que estamos representando um domínio infinito (ver 4.4), que inclui, inclusive, o conjunto dos números naturais (denotado por $\mathbb{N}$). Tal domínio incopora ainda um conjunto de mensagens ($\mathbb{M}$), o qual torna-se infinito pela possibilidade da aplicação recursiva dos operadores funcionais e pela possibilidade de uso do subscrito para os símbolos AG, NC, M e CH, além dos "índices" que indicam a execução em trânsito usados nos operadores funcionais N e K. Para resolver parte desse problema, resolvemos utilizar subconjuntos finitos dos números naturais para representar as execuções e instantes do protocolos. Dessa forma, os limites máximos de execuções e instantes passam a ser parâmetros para o sistema. Com isso, existe a possibilidade teórica de estarmos limitando as capacidades de verificação de falhas do sistema, uma vez que pode acontecer de um cenário de falha só existir quando consideradas mais execuções ou instantes do que aqueles fornecidos como parâmetros. Na prática, como demonstrado nas provas do capítulo 5, não são necessários grandes números de execuções e instantes. Contudo, dependendo da capacidade dos recursos computacionais disponíveis, nada impede que sejam usados limites tão grandes quanto se queira.

O segundo obstáculo diz respeito a possibilidade do mecanismo de prova de teoremas do PROLOG entrar num laço, mesmo existindo uma prova para a pergunta proposta. Esse é o caso de definições circulares, por exemplo, quando uma cláusula é definida em função dela mesma. Quando apresentarmos o mapeamento da $\mathcal{L}_{ep}$ para cláusulas e as cláusulas correspondentes aos axiomas de Δ_{ep}, observaremos que tais definições recursivas farão parte do sistema proposto.

Para ultrapassar a segunda parte do primeiro obstáculo juntamente com o segundo obstáculo, propomos a utilização da técnica de **busca com aprofundamento iterativo** (O'KEEFE, 1990; SHOHAM, 1994). O uso dessa técnica garante que o processo de busca do mecanismo de prova do PROLOG encerra se, e somente se, a sentença em questão for teorema da teoria que tem como axiomas as cláusulas do programa PROLOG em execução.

6.2 Passagem da $\mathcal{L}_{ep}$ para Cláusulas

Dada a impossibilidade de usarmos todos os símbolos propostos no alfabeto da $\mathcal{L}_{ep}$, vamos inicialmente mapear os termos da $\mathcal{L}_{ep}$ para cadeias de caracteres que possam ser usadas nos interpretadores PROLOG mais comuns. Algumas regras fazem uso das definições da seção 4.3.

Definição 6.1. *Para realizar a transformação sintática de um termo qualquer da $\mathcal{L}_{ep}$, siga as seguintes regras:*

1. As variáveis A, B, C, S e as constantes $1, 2, 3, ...$, $info$ e $lixo$ devem ser reescritas diretamente, sem necessidade de nenhuma transformação.

2. As constantes $1^{||}, 2^{||}, 3^{||}, ...$ devem ser transformadas para `||1`, `||2`, `||3` etc.

3. Os termos[1] E, I, M, NC, CH, AG que contêm símbolos subscritos devem ter tais subscritos substituídos por números inteiros concatenados ao seu nome. Numa mesma especificação, devem ser usados números diferentes para subscritos diferentes e números iguais para subscritos iguais. Por exemplo, a transformação do termo M_1 resulta em `M1`. Observe que não há termos desse tipo com sobrescritos.

4. Termos da forma $F(t)$, onde t é também um termo, devem ser substituídos por

   `f(`T`)`

   onde T é o resultado da transformação sintática do termo t.

5. Termos da forma $K_t, K_t^{-1}, K_{ts}, K_{ts}^r, N_t^r$ e $\overset{u}{N}{}_t^r$, onde $t, s \in \mathcal{V}_4 \cup \mathcal{C}_1$, $r \in \mathcal{C}_2 \cup \mathcal{C}_3$ e $u \in \mathcal{C}_2$, devem ser substituídos respectivamente por

   `k(`T`), k_1(`T`), k(`T`,`S`), k(`T`,`S`,`R`), nc(`T`,`R`)` e `nc(`T`,`R`,`U`)`

   onde T, S, R e U são os resultados das transformações sintáticas de t, s, r e u, respectivamente.

6. Termos da forma $\{t\}s$, onde t e s são termos, devem ser substituídos por

   `crypt(`T`,`S`)`

[1] Na verdade, os símbolos E e I, com ou sem subscritos, não são termos da $\mathcal{L}_{ep}$. Porém, para efeito de transformação sintática, podem ser tratados como tal.

onde T e S são os resultados das transformações sintáticas dos termos t e s, respectivamente.

7. Termos da forma $\langle t \rangle s$, onde t e s são termos, devem ser substituídos por

$$\texttt{decrypt}(T,S)$$

onde T e S são os resultados das transformações sintáticas dos termos t e s, respectivamente.

8. Termos da forma $[t]s$, onde t e s são termos, devem ser substituídos por

$$\texttt{ass}(T,S)$$

onde T e S são os resultados das transformações sintáticas dos termos t e s, respectivamente.

9. Termos da forma $t.s$, onde t e s são termos, devem ser substituídos por

$$T\texttt{\#}S$$

onde T e S são os resultados das transformações sintáticas dos termos t e s, respectivamente.

De forma análoga a que fizemos com os termos, vamos definir a transformação sintática das fórmulas atômicas da $\mathcal{L}_{ep}$ (ver definição 4.4). As fórmulas atômicas que não se encaixarem em nenhuma das regras não precisam ser transformadas:

Definição 6.2. *Para transformar uma fórmula atômica da $\mathcal{L}_{ep}$ siga as seguintes regras:*

1. Fórmulas atômicas da forma $s < t$, onde $s, t \in \mathcal{INS}$, devem ser substituídas por

$$S \texttt{ < } T$$

onde S e T são os resultados da transformação sintática de s e t, respectivamente.

2. Fórmulas atômicas da forma $s < t$, onde $s, t \in \mathcal{EXE}$, devem ser substituídas por

$$S \texttt{ < } T$$

onde S e T são os resultados da transformação sintática dos termos s e t, respectivamente.

3. Fórmulas atômicas da forma $s \neq t$, onde $s, t \in \mathcal{MSG}$, devem ser substituídas por

$$S \texttt{ \textbackslash= } T$$

onde S e T são os resultados da transformação sintática de s e t, respectivamente.

4. Fórmulas atômicas da forma $s \equiv t$, onde $s, t \in \mathcal{MSG}$, devem ser substituídas por

$$\texttt{equiv(}S\texttt{,}T\texttt{)}$$

onde S e T são os resultados da transformação sintática de s e t, respectivamente.

5. Fórmulas atômicas da forma $r.s\,t \rightarrow u : v$, onde $r \in \mathcal{EXE}$, $s \in \mathcal{INS}$, $t, u \in \mathcal{AGT}$ e $v \in \mathcal{MSG}$, devem ser transformadas para

$$\texttt{[}R\texttt{,}S\texttt{] envia [}T\texttt{,}U\texttt{,}V\texttt{]}$$

onde R, S, T, U e V são os resultados das transformações sintáticas de r, s, t, u e v, respectivamente.

6. Fórmulas atômicas da forma $r.s\,t_u \rightarrow v : w$, onde $r \in \mathcal{EXE}$, $s \in \mathcal{INS}$, $t, u, v \in \mathcal{AGT}$ e $w \in \mathcal{MSG}$, devem ser transformadas para

$$\texttt{[}R\texttt{,}S\texttt{] envia [}T\texttt{,}U\texttt{,}V\texttt{,}W\texttt{]}$$

onde R, S, T, U, V e W são os resultados das transformações sintáticas de r, s, t, u, v e w, respectivamente.

7. Fórmulas atômicas da forma $r.s\,t \leftarrow u : v$, onde $r \in \mathcal{EXE}$, $s \in \mathcal{INS}$, $t, u \in \mathcal{AGT}$ e $v \in \mathcal{MSG}$, devem ser transformadas para

$$\texttt{[}R\texttt{,}S\texttt{] recebe [}T\texttt{,}U\texttt{,}V\texttt{]}$$

onde R, S, T, U e V são os resultados das transformações sintáticas de r, s, t, u e v, respectivamente.

8. Fórmulas atômicas da forma $r.s\,t \leftarrow u_v : w$, onde $r \in \mathcal{EXE}$, $s \in \mathcal{INS}$, $t, u, v \in \mathcal{AGT}$ e $w \in \mathcal{MSG}$, devem ser transformadas para

$$\texttt{[}R\texttt{,}S\texttt{] recebe [}T\texttt{,}U\texttt{,}V\texttt{,}W\texttt{]}$$

onde R, S, T, U, V e W são os resultados das transformações sintáticas de r, s, t, u, v e w, respectivamente.

9. Fórmulas atômicas da forma $r.s\,t \lhd u$, onde $r \in \mathcal{EXE}$, $s \in \mathcal{INS}$, $t \in \mathcal{AGT}$ e $u \in \mathcal{MSG}$, devem ser transformadas para

$$\texttt{[}R\texttt{,}S\texttt{] possui [}T\texttt{,}U\texttt{]}$$

onde R, S, T e U são os resultados das transformações sintáticas de r, s, t e u, respectivamente.

10. Fórmulas atômicas da forma $r.s\,t \boxtimes u$, onde $r \in \mathcal{EXE}$, $s \in \mathcal{INS}$, $t \in \mathcal{AGT}$ e $u \in \mathcal{MSG}$, devem ser transformadas para

$$\texttt{[}R\texttt{,}S\texttt{] armazena [}T\texttt{,}U\texttt{]}$$

onde R, S, T e U são os resultados das transformações sintáticas de r, s, t e u, respectivamente.

11. Fórmulas atômicas da forma $r.s\,t \overset{v}{\sim} u$, onde $r \in \mathcal{EXE}$, $s \in \mathcal{INS}$, $t, u \in \mathcal{AGT}$ e $v \in \mathcal{MSG}$, devem ser transformadas para

$$\texttt{[}R\texttt{,}S\texttt{] relaciona [}T\texttt{,}V\texttt{,}U\texttt{]}$$

onde R, S, T, U e V são os resultados das transformações sintáticas de r, s, t, u e v, respectivamente.

12. Fórmulas atômicas da forma $r.s\,t \overset{v}{\leftrightarrow} u$, onde $r \in \mathcal{EXE}$, $s \in \mathcal{INS}$, $t, u \in \mathcal{AGT}$ e $v \in \mathcal{MSG}$, devem ser transformadas para

$$\texttt{[}R\texttt{,}S\texttt{] chaveboa [}T\texttt{,}V\texttt{,}U\texttt{]}$$

onde R, S, T, U e V são os resultados das transformações sintáticas de r, s, t, u e v, respectivamente.

Para usarmos cláusulas como

```
[E,I] envia [A,s,a#nc(a,E)]
```

tivemos que definir cada um dos novos operadores infixos descritos acima. Os operadores são definidos com aridade 2 e cada um de seus argumentos é uma lista PROLOG (ver apêndice C).

Finalmente, podemos definir a transformação das fórmulas da $\mathcal{L}_{ep}$:

Definição 6.3. *Para transformar uma fórmula da $\mathcal{L}_{ep}$ siga as seguintes regras:*

1. Fórmulas do tipo $\varphi_1, \varphi_2, \ldots, \varphi_n$, onde $\varphi_1, \varphi_2, \ldots, \varphi_n$ são fórmulas atômicas, ou seja, fórmulas do tipo conjunção, devem ser substituídas por n cláusulas do tipo:

$$T_i\,.$$

   onde T_i, $1 \leq i \leq n$, é o resultado da transformação sintática da fórmula atômica φ_i.

2. Fórmulas da forma $\varphi_1, \varphi_2, \ldots, \varphi_n \Leftarrow \psi_1, \psi_2, \ldots, \psi_m$, onde φ_i, $1 \leq i \leq n$, e ψ_j, $1 \leq j \leq m$, são fórmulas atômicas, devem ser transformadas em n cláusulas do tipo:

$$T_i \texttt{:-}\ S_1, S_2, \ldots, S_m.$$

   onde T_i, $1 \leq i \leq n$, e S_j, $1 \leq j \leq m$ são os resultados das transformações sintáticas de cada fórmula atômica φ_i e ψ_j, respectivamente.

3. Fórmulas da forma $\varphi_1, \varphi_2, \ldots, \varphi_n \Leftrightarrow \psi_1, \psi_2, \ldots, \psi_m$, onde φ_i, $1 \leq i \leq n$, e ψ_j, $1 \leq j \leq m$, são fórmulas atômicas, devem ser transformadas em $n+m$ cláusulas do tipo:

$$\begin{aligned} T_i &\texttt{:-}\ S_1, S_2, \ldots, S_m. \\ S_j &\texttt{:-}\ T_1, T_2, \ldots, T_n. \end{aligned}$$

   onde T_i, $1 \leq i \leq n$, e S_j, $1 \leq j \leq m$, são os resultados das transformações sintáticas de cada fórmula atômica φ_i e ψ_j, respectivamente.

6.3 Cláusulas que Representam os Axiomas do Ambiente

Nesta seção, vamos apresentar a transformação sintática de alguns dos axiomas que representam o ambiente de comunicação (ver 4.5). Antes de iniciarmos, vamos discutir como representar as limitações no domínio dos números naturais e a **tipagem** da $\mathcal{L}_{ep}$.

Para limitar o número de instantes com que se deseja proceder a análise, basta declarar os possíveis valores para as variáveis do tipo `I` (com ou sem número seguindo, ou seja, variáveis como: `I1`, `I2` etc), as quais representam instantes. Para isso, criamos o operador básico `instante`. Por exemplo, para limitar uma variável do tipo `I` aos valores 1,2 e 3, basta declarar

```
instante(1). instante(2). instante(3).
```

e na cláusula onde `I` está presente, adicionar `instante(I)` como condição. Continuando com o exemplo, uma cláusula da forma

```
[E,I] envia [a,b,a#b#nc(a,E)]
```

seria então expandida para

```
[E,I] envia [a,b,a#b#nc(a,E)]:-
    instante(I).
```

Dessa forma, diminuímos o espaço de busca por uma prova que use essa cláusula, restringindo-o àqueles ramos da árvore de prova nos quais a variável do tipo `I` é igual a 1, 2 ou 3.

O mesmo tipo de extensão deve ser usado para variáveis do tipo execução, agentes e mensagens, representado então a tipagem proposta nas seções 4.3 e 4.4. Para isso, criamos os operadores `execucao`, `agentes` e `mensagem`. O operador `execucao` é usado de forma análoga a do `instante`, ou seja, basta declarar os possíveis valores para as execuções e expandir as cláusulas que fazem referência a execuções. No caso do operador `agente`, as declarações dos valores possíveis para as variáveis do tipo agente não ficam por conta do usuário (a não ser que ele deseje estender a $\mathcal{L}_{ep}$), e são fixadas no sistema como

```
agente(a). agente(b). agente(c). agente(s).
```

restando somente expandir as cláusulas que contêm variáveis desse tipo. As declarações dos valores das variáveis do tipo mensagem também são fixadas no sistema. Elas representam exatamente as possibilidades de criação de termos como proposto na definição 4.3, ou seja, o conjunto de valores possíveis é infinito, dada a possibilidade da aplicação recursiva das regras de formação de termos. A codificação para os valores das mensagens ficou definida como descrito abaixo.

```
mensagem(A):- % identidades dos agentes
    agente(A).
mensagem(k(A)):- % chaves publicas
    agente(A).
mensagem(k_1(A)):- % chaves privadas
    agente(A).
mensagem(k(A,B)):- % chaves compartilhadas
    agente(A),
    agente(B).
mensagem(k(A,B,E)):- % chaves de sessao
    agente(A),
    agente(B),
    execucao(E).
mensagem(nc(A,E)):- % nonces
    agente(A),
    execucao(E).
mensagem(M1#M2):- % concatenacao de mensagens
    mensagem(M1),
    mensagem(M2).
mensagem(crypt(M,CH)):- % mensagens criptografadas
    mensagem(M),
    mensagem(CH).
mensagem(ass(M,CH)):- % mensagens assinadas
    mensagem(M),
    mensagem(CH).
mensagem(decrypt(M,CH)):- % resultado da descriptografia
    mensagem(M),
    mensagem(CH).
mensagem(f(M)):- % funcao de modificacao
    mensagem(M).
```

Uma vez que os axiomas que envolvem as relações de envio e recebimento de mensagens estão definidos para o caso genérico da personificação, precisamos incluir no sistema de cláusulas a conversão para o caso em que um agente passa-se por ele mesmo, permitindo assim que as especificações e consultas por falhas sejam inseridas sem necessariamente considerar a possibilidade de personificação. As cláusulas responsáveis pela referida transformação são:

```
[E,I] envia [A,A,B,M]:-
    [E,I] envia [A,B,M].

[E,I] envia [A,B,M]:-
    [E,I] envia [A,A,B,M].

[E,I] recebe [A,B,B,M]:-
    [E,I] recebe [A,B,M].

[E,I] recebe [A,B,M]:-
    [E,I] recebe [A,B,B,M].
```

A seguir, são apresentadas as cláusulas que resultaram da transformação sintática de alguns axiomas do conjunto Δ_{ep}. Perceba que as cláusulas já foram expandidas para representar a tipagem da $\mathcal{L}_{ep}$. Por motivos didáticos, mostraremos apenas as cláusulas correspondentes aos axiomas que foram usados na prova do teorema 5.1, o qual demonstra formalmente a falha no protocolo de Nesset discutida na seção 5.2.

Canal Visível (CV)
$E.I_2\, A \leftarrow C_S : M \Leftarrow$
$E.I_1\, C_S \rightarrow B : M,$
$I_1 < I_2$
	`% Canal Visível`
	`[E,I2] recebe [A,C,S,M] :-`
	`execucao(E),`
	`instante(I1), instante(I2),`
	`agente(A),agente(B),agente(C),`
	`mensagem(M),`
	`I1 < I2,`
	`[E,I1] envia [C,S,B,M].`

Possui se Recebe (PR)
$E.I\, A \lhd M \Leftarrow$
$E.I\, A \leftarrow C_B : M$
	`% Possui se Recebe`
	`[E,I] possui [A,M]:-`
	`execucao(E),`
	`instante(I),`
	`agente(A),`
	`mensagem(M),`
	`[E,I] recebe [A,C,B,M].`

Mensagem Assinada (MA)
$E.I\ A \lhd M \Leftarrow$
$E.I\ A \lhd [M]K_B^{-1}$,
$E.I\ A \lhd K_B$
	`% Mensagem Assinada`
	`[E,I] possui [A,M]:-`
	`execucao(E),`
	`instante(I),`
	`agente(A),`
	`mensagem(M),`
	`[E,I] possui [A,ass(M,k_1(B))],`
	`[E,I] possui [A,k(B)].`

Possui Chave Pública (PCB)
$E.I\ A \lhd K_B$
	`% Possui Chave Pública`
	`[E,I] possui [A,k(B)]:-`
	`execucao(E),`
	`instante(I),`
	`agente(A), agente(B).`

Mensagem Simples (MS)
$E.I\ A \lhd M_1.M_2 \Leftrightarrow$
$E.I\ A \lhd M_1$,
$E.I\ A \lhd M_2$
	`% Mensagem Simples`
	`[E,I] possui [A,M1#M2]:-`
	`execucao(E),`
	`instante(I),`
	`agente(A),`
	`mensagem(M1),`
	`mensagem(M2),`
	`[E,I] possui [A,M1],`
	`[E,I] possui [A,M2].`
	`[E,I] possui [A,M1]:-`
	`execucao(E),`
	`instante(I),`
	`agente(A),`
	`mensagem(M1),`
	`mensagem(M2),`
	`[E,I] possui [A,M1#M2].`
	`[E,I] possui [A,M2]:-`
	`execucao(E),`
	`instante(I),`
	`agente(A),`
	`mensagem(M1),`
	`mensagem(M2),`
	`[E,I] possui [A,M1#M2].`

Os axiomas MQ e IU serão implementados no sistema diretamente através dos predicados nativos do PROLOG `<` e `\=`, respectivamente.

6.4 O Meta-Interpretador

Para contornar a possibilidade de termos laços intermináveis nas cláusulas resultantes da transformação sintática das sentenças da $\mathcal{L}_{ep}$ da especificação do protocolo e dos axiomas de ambiente, utilizamos a técnica de busca com aprofundamento iterativo (O'KEEFE, 1990; SHOHAM, 1994). Tal técnica consiste na alteração do comportamento do mecanismo de busca do PROLOG. Em vez de utilizar simplesmente a busca em profundidade sem limite de níveis, uma busca em profundidade limitada a um nível é realizada antes que tal nível seja incrementado e a busca continue. Na verdade, quando o limite de profundidade é incrementado, todo o espaço de busca dos limites anteriores é novamente pesquisado. Essa é uma desvantagem da utilização dessa técnica, porém, é o preço que devemos pagar para poder usar cláusulas com declarações circulares. Através da utilização dessa técnica, temos ainda a garantia de que a busca por uma prova de uma falha acaba, se e somente se, a fórmula que representa tal falha for realmente teorema da teoria formada pela especificação do protocolo e os axiomas de ambiente.

Apresentamos abaixo o código PROLOG que implementa a técnica de aprofundamento iterativo.

```
prova(true,_,_):- !.
prova(_,D,Limite):-
    D > Limite,
    !,
    fail.  %% alcan\c{c}ou a profundidade limite.
prova((A,B),D,Limite):-
    !,
    prova(A,D,Limite),
    prova(B,D,Limite).
prova(A,_,_):-
    predicate_property(A,built_in),
    !,
    call(A).
prova(A,D,Limite):-
    clause(A,B),
    D1 is D+1,
    prova(B,D1,Limite).
aprofundamento_iterativo(G,D):-
    prova(G,0,D).
aprofundamento_iterativo(G,D):-
    write('limite='),
    write(D),
    write(' alcan\c{c}ado. '),
    write('(Tecle <Enter> para continuar.)'),
    get0(C),
    ( C == 10 ->
        D1 is D + 1,
        aprofundamento_iterativo(G,D1)).
```

Esse código pode ser usado para realizar a busca com aprofundamento iterativo em qualquer programa PROLOG. Para utilizá-lo, basta consultar o predicado

```
aprofundamento_iterativo
```

utilizando como primeiro argumento a cláusula a ser provada e como segundo argumento a profundidade inicial de busca. Caso a profundidade inicial seja alcançada sem sucesso na busca, o sistema solicitará ao analista sua confirmação para avançar na busca utilizando a cada confirmação uma profundidade 1 nível superior à anterior.

É importante que todos os axiomas necessários para a prova de uma determinada falha estejam codificados, sob pena de uma prova de tal falha não ser encontrada, e portanto, da busca nunca acabar.

6.5 Exemplo: O Protocolo de Nesset

Nesta seção, apresentamos o código resultante da transformação sintática do protocolo de Nesset, analisado formalmente na seção 5.2. Estas cláusulas dever ser colocadas juntas com as cláusulas dos axiomas de ambiente (ver 6.3) e com o meta-interpretador.

```
[E,I] envia [A,B,ass(nc(A,E)#k(A,B,E),k_1(A))]:-
    execucao(E),
    instante(I),
    agente(A),
    agente(B).

[E,I] envia [B,A,crypt(nc(B,E),CH)]:-
   execucao(E),
   instante(I),
   agente(A),
   agente(B),
   I2 < 1,
   [E,I2] recebe [B,A,ass(_#CH,k_1(A))].
```

6.6 Conclusões e Resultados

Na tentativa de demonstrar automaticamente o cenário de ataque descrito formalmente pela prova do teorema 5.1, submetemos a seguinte consulta a nosso sistema:

```
aprofundamento_iterativo(([E,I] possui [C,k(A,B,1)], C \= A, C \= B),9).
```

Infelizmente, não obtivemos resposta em tempo hábil. Realizando testes com as cláusulas que definem as possíveis mensagens, percebemos que o tamanho espaço de busca sendo pesquisado explica a demora por respostas. Decidimos modificar a definição das mensagens de nosso sistema de modo a reduzir as possibilidades de formação de mensagens. Tais modificações permitem parametrizar propriedades das mensagens, a saber, o número máximo de submensagens que podem formar uma mensagem maior, o número máximo de aplicações aninhadas do operador `ass()` e número máximo de aplicações aninhadas do operador `crypt()`. Uma vez que já conhecíamos o cenário de ataque buscado e o formato das mensagens que o compõe, estabelecemos como limites a aplicação de somente 1 nível para cada um desses operadores. A redefinição da formação de possíveis mensagens ficou assim descrita:

```
limites_mensagens([1,1,1]). % #, crypt() e ass(), respectivamente

mensagem(M):-
    limites_mensagens([LimSubMsg,LimCryp,LimAss]),
    mensagem(M,[LimSubMsg,LimCryp,LimAss]).

mensagem(X,_):-
    agente(X).

mensagem(k(X),_):-
   agente(X).

mensagem(k_1(X),_):-
    agente(X).

mensagem(k(A,B,E),_):-
    agente(A),
    agente(B),
    execucao(E).

mensagem(nc(A,E),_):-
    agente(A),
    execucao(E).

mensagem(M1#M2,[LimSubMsg,LimCryp,LimAss]):-
    LimSubMsg > 0,
    NovoLimSubMsg is LimSubMsg - 1,
    mensagem(M1,[0,LimCryp,LimAss]),
    mensagem(M2,[NovoLimSubMsg,LimCryp,LimAss]).

mensagem(ass(M,CH),[LimSubMsg,LimCryp,LimAss]):-
    LimAss > 0,
    NovoLimAss is LimAss - 1,
    mensagem(M,[LimSubMsg,LimCryp,NovoLimAss]),
    mensagem(CH,[LimSubMsg,LimCryp,0]).

mensagem(crypt(M,CH),[LimSubMsg,LimCryp,LimAss]):-
   LimCryp > 0,
   NovoLimCryp is LimCryp - 1,
   mensagem(M,[LimSubMsg,NovoLimCryp,LimAss]),
   mensagem(CH,[LimSubMsg,0,LimAss]).
```

Note que a possibilidade de formação de mensagens do tipo `decrypt()` e `f()` foi eliminada.

Esperávamos com isso poder fornecer uma resposta parametrizada, ou seja, com restrições de formato de mensagens, em tempo hábil. Essas modificações não surtiram o efeito desejado, pois, mesmo para parâmetros mínimos para encontrar a prova desejada, o número de combinações de mensagens possíveis, combinado à possibilidade de aplicação recursiva das regras de formação de mensagens simples (axioma MS), ainda é proibitivo.

Como última tentativa, resolvemos excluir algumas cláusulas que sabemos não ser necessárias para a demonstração da falha procurada, limitando assim ainda mais o espaço de busca.

Foram retiradas as seguintes cláusulas que permitem a formação de mensagens:

```
mensagem(k(X),_):-
   agente(X).

mensagem(crypt(M,CH),[LimSubMsg,LimCryp,LimAss]):-
   LimCryp > 0,
   NovoLimCryp is LimCryp - 1,
   mensagem(M,[LimSubMsg,NovoLimCryp,LimAss]),
   mensagem(CH,[LimSubMsg,0,LimAss]).

[E,I] possui [A,M1]:-
    execucao(E),
    instante(I),
    agente(A),
    mensagem(M1),
    [E,I] possui [A,M1#_].

[E,I] possui [A,M1#M2]:-
    execucao(E),
    agente(A),
    mensagem(M1),
    mensagem(M2),
    [E,I] possui [A,M1],
    [E,I] possui [A,M2].
```

Além dessas, a cláusula que representa o segundo passo do protocolo, descrita abaixo, também foi eliminada.

```
[E,I] envia [B,A,crypt(nc(B,E),CH)]:-
   execucao(E),
   instante(I),
   agente(A),
   agente(B),
   I2 < 1,
   [E,I2] recebe [B,A,ass(_#CH,k_1(A))].
```

Dessa vez, obtivemos diversas respostas interessantes, todas descritas no anexo C. Dentre elas, estão algumas que não oferecem um significado razoável, tais como aquelas que consideram uma execução onde um mesmo agente é o iniciador e o respondedor, simultaneamente, como se pode ver abaixo.

```
E = 1 I = 2 C = a A = b B = b ;

E = 1 I = 2 C = a A = c B = c ;
```

Esse tipo de resposta, embora não signifique diretamente uma falha no contexto da análise do protocolo de Nesset, é resultado esperado do tipo de análise proposto, o qual busca explorar possibilidades diversas de personificação, substituição de mensagens e identidades etc . Cabe ao analista avaliar cada caso e reconhecer em cada resposta fornecida a caracterização, ou não, de uma falha.

Entretanto, algumas respostas representam exatamente o resultado dos cenários de ataque onde um terceiro agente, que não o iniciador ou o respondedor, chega a possuir a chave de sessão proposta pelo iniciador. Esse é o caso das respostas abaixo.

```
E = 1 I = 2 C = b A = c B = a ;

E = 1 I = 2 C = c A = a B = b ;
```

Apesar desse último não ser um cenário realista, tendo sido moldado exclusivamente para a análise de um determinado protocolo, os resultados obtidos demonstram a funcionalidade, mesmo que de limitada, da abordagem seguida. Resta o consolo de que talvez não tenhamos fornecido os recursos computacionais e o tempo necessários para que o tipo de processamento simbólico efetuado nos dois primeiros cenários fosse concluído em um tempo factível.

Conclui-se assim que, mesmo para um protocolo curto e com uma falha bem conhecida, cuja demonstração "manual" não requer tantos passos, a demonstração automática para

o caso geral não oferece resultados plausíveis em tempo hábil. Contudo, essa limitação extrapola o escopo deste trabalho, sendo uma característica comum aos métodos de prova automática de teoremas para as lógicas de primeira ordem.

7 CONCLUSÕES GERAIS

Diversas propostas de protocolos para autenticação e distribuição de chaves já foram incorporadas a tecnologias (OPPLIGER, 1996). Algumas delas, mesmo problemáticas, permaneceram consideradas como seguras por algum tempo (SCHNEIER, 1996; OPPLIGER, 1996; KAUFMAN; PERLMAN; SPECINER, 1995; HUGHES, 1995; STALLINGS, 1995; CHESWICK; BELLOVIN, 1994; CHAPMAN; ZWICKY, 1995). À medida que novas formas de ataque, como os descritas nos capítulos 2 e 5, vão surgindo, a demanda por métodos formais eficazes aumenta. A experiência mostra ainda que, além da eficácia, os métodos devem possuir outras características importantes, tais como a eficiência em apresentar respostas em tempo viável e a propriedade de tratar o processo de desenvolvimento como um todo, diminuindo a distância entre a teoria e a real implementação dos protocolos. Uma tendência apontada em Carlsen (1994b, p. 59), é a de usar métodos cada vez mais específicos para detecção de classes bem definidas de falhas e submeter um protocolo aos vários processos de análise em seqüência. Apesar de simpatizarmos com essa abordagem, procuramos atender, além das duas características citadas acima, um escopo maior de protocolos e falhas.

Foi com esses objetivos em mente que iniciamos nossa pesquisa. Como principais resultados, podemos citar:

- A demonstração, de acordo com as seções 2.5, 2.6 e 4.6, de que o método de especificação tradicionalmente usado na descrição dos protocolos, conhecido como notação padrão, é insuficiente para servir como ferramenta de análise sob vários aspectos, a saber:

    1. Não-representação das suposições iniciais;
    2. A inabilidade em representar a dependência entre valores de mensagens recebidas e enviadas;
    3. A ambigüidade na descrição de quais chaves cada agente deve utilizar em cada processo de criptografia e descriptografia;

4. Não-representação das ações internas;
5. Não-representação das alterações no estado de conhecimento dos agentes, mesmo que parcial;
6. Não-representação dos objetivos finais.

- Uma proposta (na verdade uma nova abordagem) de método formal baseada numa linguagem de primeira ordem e nos métodos de prova de teoremas. Tal proposta apresenta as seguintes contribuições:

  1. Uma linguagem de especificação formal capaz de superar as principais deficiências da notação padrão;
  2. A representação formal, através de um sistema de axiomas, do ambiente que envolve os protocolos de criptografia, especificando inclusive as capacidades dos agentes malignos;
  3. A possibilidade da demonstração formal de falhas previamente conhecidas, inclusive com a apresentação do cenário de ataque onde tal falha ocorre;
  4. A possibilidade de verificação formal de uma possível falha, também com a apresentação de um cenário de ataque, caso a falha seja provada.

A automatização do método proposto agilizaria a verificação de possíveis falhas, livrando o usuário das demonstrações manuais (como aquelas do apêndice A). Além disso, permitiria a verificação de variações no cenário de ataque de uma falha previamente conhecida.

Infelizmente, o processo de automatização da demonstração de falhas, proposto no capítulo 6, não apresentou resultados em tempo viável para o protocolo e a falha analisados, limitando-se a fornecer respostas em tempo viável somente quando retiramos cláusulas não utilizadas para a demonstração em questão, ou seja, quando diminuímos o espaço de busca. Pela simplicidade do exemplo analisado, somos levados a concluir que o mesmo tipo de problema aconteceria na análise automatizada de protocolos e falhas mais elaborados. Resta-nós o consolo de que, em sendo uma possível falha provável a partir de uma especificação proposta, então temos garantido o fato de que, em algum momento, o processo de prova automática proposto termina.

7.1 Trabalhos Futuros

Como possibilidades de extensão a este trabalho, citamos:

1. Aumento do escopo da linguagem de especificação, podendo representar diferentes sistemas criptográficos (simétricos e assimétricos) e suas propriedades particulares. Ou ainda, novas classes de protocolos, além dos de autenticação e distribuição de chaves, tais como aqueles em que as chaves são compartilhadas entre mais de dois agentes;

2. Investigação do uso da modalidade para representar todos os componentes do conhecer, como discutido na seção 4.2, possibilitando a representação formal de todas as possíveis modificações no estado de conhecimento dos agentes.

APÊNDICE A – PROVAS COMPLETAS

Listamos aqui as provas completas dos teoremas enunciados no capítulo 5, as quais não foram inteiramente apresentadas por serem demasiadamente extensas.

A.1 Teoremas sobre o Protocolo Needham-Schroeder

Teorema 5.2, página 104:

$$\begin{aligned}\Phi_{ns} \cup \Delta_{ep} \vdash\ & \exists\ E, I_1, I_2 \\ & E.I_1\, a \overset{K^E_{ab}}{\leftrightarrow} b, \\ & E.I_2\, b \overset{K^E_{ab}}{\leftrightarrow} a\end{aligned}$$

Demonstração.

D1	Γ P1	suposição
D2	Γ $1.1\, a \boxtimes N^1_a,\ 1.1\, a \overset{N^1_a}{\sim} b$	D1, substituição
D3	Γ P2	suposição
D4	Γ $1.1\, a \rightarrow s : a.b.N^1_a \Leftarrow$ $1.1\, a \boxtimes N^1_a,$ $1.1\, a \overset{N^1_a}{\sim} b$	D3, substituição
D5	Γ $1.1\, a \rightarrow s : a.b.N^1_a$	D2, D4, modus-ponens
D6	Γ TN	suposição
D7	Γ $1.2\, s \leftarrow a : a.b.N^1_a \Leftarrow$ $1.1\, a \rightarrow s : a.b.N^1_a,$ $1 < 2$	D6, substituição
D8	Γ MQ	suposição
D9	Γ $1 < 2$	D8, substituição
D10	Γ $1.1\, a \rightarrow s : a.b.N^1_a,\ 1 < 2$	D5, D9, conjunção
D11	Γ $1.2\, s \leftarrow a : a.b.N^1_a$	D10, D7, modus-ponens

D12	Γ SI2	suposição
D13	Γ $1.3\, s \lhd K_{as}$	D12, substituição
D14	Γ SI4	suposição
D15	Γ $1.3\, s \lhd K_{bs}$	D14, substituição
D16	Γ SI5	suposição
D17	Γ $1.3\, s \lhd K^1_{ab}$	D16, substituição
D18	Γ $2 < 3$	D8, substituição
D19	Γ $1.2\, s \leftarrow a : a.b.N^1_a,\ 1.3\, s \lhd K_{as}$	D11, D13, conjunção
D20	Γ $1.2\, s \leftarrow a : a.b.N^1_a$, $1.3\, s \lhd K_{as}$, $1.3\, s \lhd K_{bs}$	D19, D15, conjunção
D21	Γ $1.2\, s \leftarrow a : a.b.N^1_a$, $1.3\, s \lhd K_{as}$, $1.3\, s \lhd K_{bs}$, $1.3\, s \lhd K^1_{ab}$	D20, D17, conjunção
D22	Γ $1.2\, s \leftarrow a : a.b.N^1_a$, $1.3\, s \lhd K_{as}$, $1.3\, s \lhd K_{bs}$, $1.3\, s \lhd K^1_{ab}$, $2 < 3$	D21, D18, conjunção
D23	Γ P3	suposição
D24	Γ $1.3\, s \rightarrow a : \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as} \Leftarrow$ $1.2\, s \leftarrow a : a.b.N^1_a$, $1.3\, s \lhd K_{as}$, $1.3\, s \lhd K_{bs}$, $1.3\, s \lhd K^1_{ab}$, $2 < 3$	D23, substituição
D25	Γ $1.3\, s \rightarrow a : \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as}$	D22, D24, modus-ponens
D26	Γ $1.4\, a \leftarrow s : \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as} \Leftarrow$ $1.3\, s \rightarrow a : \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as}$, $3 < 4$	D6, substituição
D27	Γ $3 < 4$	D8, substituição
D28	Γ $1.3\, s \rightarrow a : \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as},\ 3 < 4$	D25, D27, conjunção
D29	Γ $1.4\, a \leftarrow s : \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as}$	D28, D26, modus-ponens
D30	Γ SI1	suposição
D31	Γ $1.5\, a \lhd K_{as}$	D30, substituição
D32	Γ $1.4\, a \leftarrow s : \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as}$, $1.5\, a \lhd K_{as}$	D29, D31, conjunção

D33	$\Gamma\ 4 < 5$	D8, substituição
D34	$\Gamma\ 1 < 4$	D8, substituição
D35	$\Gamma\ 1.4\, a \leftarrow s : \{N_a^1.b.K_{ab}^1.\{K_{ab}^1.a\}K_{bs}\}K_{as}$, $1.5\, a \lhd K_{as}$, $1.1\, a \boxtimes N_a^1$, $1.1\, a \overset{N_a^1}{\sim} b$	D32, D2, conjunção
D36	$\Gamma\ 1.4\, a \leftarrow s : \{N_a^1.b.K_{ab}^1.\{K_{ab}^1.a\}K_{bs}\}K_{as}$, $1.5\, a \lhd K_{as}$, $1.1\, a \boxtimes N_a^1$, $1.1\, a \overset{N_a^1}{\sim} b$, $4 < 5$	D35, D33, conjunção
D37	$\Gamma\ 1.4\, a \leftarrow s : \{N_a^1.b.K_{ab}^1.\{K_{ab}^1.a\}K_{bs}\}K_{as}$, $1.5\, a \lhd K_{as}$, $1.1\, a \boxtimes N_a^1$, $1.1\, a \overset{N_a^1}{\sim} b$, $4 < 5$, $1 < 4$	D36, D34, conjunção
D38	Γ P4	suposição
D39	$\Gamma\ 1.5\, a \rightarrow b : \{K_{ab}^1.a\}K_{bs}, 1.5\, a \overset{K_{ab}^1}{\leftrightarrow} b \Leftarrow$ $1.4\, a \leftarrow s : \{N_a^1.b.K_{ab}^1.\{K_{ab}^1.a\}K_{bs}\}K_{as}$, $1.5\, a \lhd K_{as}$, $1.1\, a \boxtimes N_a^1$, $1.1\, a \overset{N_a^1}{\sim} b$, $4 < 5$, $1 < 4$	D38, substituição
D40	$\Gamma\ 1.5\, a \rightarrow b : \{K_{ab}^1.a\}K_{bs},\ 1.5\, a \overset{K_{ab}^1}{\leftrightarrow} b$	D37, D39, modus-ponens
D41	$\Gamma\ 1.5\, a \rightarrow b : \{K_{ab}^1.a\}K_{bs}$	D40, conjunção
D42	$\Gamma\ 1.6\, b \leftarrow a : \{K_{ab}^1.a\}K_{bs} \Leftarrow$ $1.5\, a \rightarrow b : \{K_{ab}^1.a\}K_{bs}$, $5 < 6$	D6, substituição
D43	$\Gamma\ 5 < 6$	D8, substituição
D44	$\Gamma\ 1.5\, a \rightarrow b : \{K_{ab}^1.a\}K_{bs},\ 5 < 6$	D41, D43, conjunção
D45	$\Gamma\ 1.6\, b \leftarrow a : \{K_{ab}^1.a\}K_{bs}$	D44, D42, modus-ponens
D46	Γ SI3	suposição
D47	$\Gamma\ 1.7\, b \lhd K_{bs}$	D46, substituição
D48	$\Gamma\ 1.6\, b \leftarrow a : \{K_{ab}^1.a\}K_{bs},\ 1.7\, b \lhd K_{bs}$	D45, D47, conjunção
D49	$\Gamma\ 6 < 7$	D8, substituição

D50	$\Gamma\ 1.6\, b \leftarrow a : \{K^1_{ab}.a\}K_{bs},\ 1.7\, b \lhd K_{bs},\ 6 < 7$	D48, D49, conjunção
D51	Γ P5	suposição
D52	$\Gamma\ 1.7\, b \rightarrow a : \{N^1_b\}K^1_{ab},$ $1.7\, b \boxtimes F(N^1_b),$ $1.7\, b \overset{F(N^1_b)}{\sim} a \Leftarrow$ $1.6\, b \leftarrow a : \{K^1_{ab}.a\}K_{bs}, 1.7\, b \lhd K_{bs}, 6 < 7$	D51, substituição
D53	$\Gamma\ 1.7\, b \rightarrow a : \{N^1_b\}K^1_{ab},$ $1.7\, b \boxtimes F(N^1_b),$ $1.7\, b \overset{F(N^1_b)}{\sim} a$	D50, D52, modus-ponens
D54	$\Gamma\ 1.7\, b \rightarrow a : \{N^1_b\}K^1_{ab}$	D53, conjunção
D55	$\Gamma\ 1.8\, a \leftarrow b : \{N^1_b\}K^1_{ab} \Leftarrow$ $1.7\, b \rightarrow a : \{N^1_b\}K^1_{ab},$ $7 < 8$	D6, substituição
D56	$\Gamma\ 7 < 8$	D8, substituição
D57	$\Gamma\ 1.7\, b \rightarrow a : \{N^1_b\}K^1_{ab},\ 7 < 8$	D54, D56, conjunção
D58	$\Gamma\ 1.8\, a \leftarrow b : \{N^1_b\}K^1_{ab}$	D57, D55, modus-ponens
D59	Γ PR	suposição
D60	$\Gamma\ 1.4\, a \lhd \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as} \Leftarrow$ $1.4\, a \leftarrow s : \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as}$	D59, substituição
D61	$\Gamma\ 1.4\, a \lhd \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as}$	D29, D60, modus-ponens
D62	Γ PIP	suposição
D63	$\Gamma\ 1.8\, a \lhd \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as} \Leftarrow$ $1.4\, a \lhd \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as},$ $4 < 8$	D62, substituição
D64	$\Gamma\ 4 < 8$	D8, substituição
D65	$\Gamma\ 1.4\, a \lhd \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as},\ 4 < 8$	D61, D64, conjunção
D66	$\Gamma\ 1.8\, a \lhd \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as}$	D65, D63, modus-ponens
D67	$\Gamma\ 1.8\, a \lhd K_{as} \Leftarrow$ $1.5\, a \lhd K_{as},$ $5 < 8$	D62, substituição
D68	$5 < 8$	D8, substituição
D69	$\Gamma\ 1.5\, a \lhd K_{as},\ 5 < 8$	D31, D68, conjunção
D70	$\Gamma\ 1.8\, a \lhd K_{as}$	D69, D67, modus-ponens
D71	$\Gamma\ 1.8\, a \lhd \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as},$ $1.8\, a \lhd K_{as}$	D66, D70, conjunção
D72	Γ MCC	suposição
D73	$\Gamma\ 1.8\, a \lhd N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs} \Leftarrow$	D72, substituição

	$1.8\,a \lhd \{N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}\}K_{as}$,	
	$1.8\,a \lhd K_{as}$	
D74	$\Gamma\ 1.8\,a \lhd N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}$	D71, D73, modus-ponens
D75	Γ MS	suposição
D76	$\Gamma\ 1.8\,a \lhd N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs} \Leftrightarrow$	D75, substituição
	$1.8\,a \lhd N^1_a.b.K^1_{ab}, 1.8\,a \lhd \{K^1_{ab}.a\}K_{bs}$	
D77	$\Gamma\ 1.8\,a \lhd N^1_a.b.K^1_{ab}, 1.8\,a \lhd \{K^1_{ab}.a\}K_{bs} \Leftarrow$	D76, bi-implicação
	$1.8\,a \lhd N^1_a.b.K^1_{ab}.\{K^1_{ab}.a\}K_{bs}$	
D78	$\Gamma\ 1.8\,a \lhd N^1_a.b.K^1_{ab},\ 1.8\,a \lhd \{K^1_{ab}.a\}K_{bs}$	D74, D77, modus-ponens
D79	$\Gamma\ 1.8\,a \lhd N^1_a.b.K^1_{ab}$	D78, conjunção
D80	$\Gamma\ 1.8\,a \lhd N^1_a.b.K^1_{ab} \Leftrightarrow$	D75, substituição
	$1.8\,a \lhd N^1_a.b,\ 1.8\,a \lhd K^1_{ab}$	
D81	$\Gamma\ 1.8\,a \lhd N^1_a.b, 1.8\,a \lhd K^1_{ab} \Leftarrow$	D80, bi-implicação
	$1.8\,a \lhd N^1_a.b.K^1_{ab}$	
D82	$\Gamma\ 1.8\,a \lhd N^1_a.b,\ 1.8\,a \lhd K^1_{ab}$	D79, D81, modus-ponens
D83	$\Gamma\ 1.8\,a \lhd K^1_{ab}$	D82, conjunção
D84	$\Gamma\ 8 < 9$	D8, substituição
D85	$\Gamma\ 1.8\,a \leftarrow b : \{N^1_b\}K^1_{ab},\ 1.8\,a \lhd K^1_{ab}$	D58, D83, conjunção
D86	$\Gamma\ 1.8\,a \leftarrow b : \{N^1_b\}K^1_{ab},\ 1.8\,a \lhd K^1_{ab},\ 8 < 9$	D85, D84, conjunção
D87	Γ P6	suposição
D88	$\Gamma\ 1.9\,a \rightarrow b : \{F(N^1_b)\}K^1_{ab} \Leftarrow$	D87, substituição
	$1.8\,a \leftarrow b : \{N^1_b\}K^1_{ab},\ 1.8\,a \lhd K^1_{ab},\ 8 < 9$	
D89	$\Gamma\ 1.9\,a \rightarrow b : \{F(N^1_b)\}K^1_{ab}$	D86, D88, modus-ponens
D90	$\Gamma\ 1.10\,b \leftarrow a : \{F(N^1_b)\}K^1_{ab} \Leftarrow$	D6, substituição
	$1.9\,a \rightarrow b : \{F(N^1_b)\}K^1_{ab}$,	
	$9 < 10$	
D91	$\Gamma\ 9 < 10$	D8, substituição
D92	$\Gamma\ 1.9\,a \rightarrow b : \{F(N^1_b)\}K^1_{ab},\ 9 < 10$	D89, D91, conjunção
D93	$\Gamma\ 1.10\,b \leftarrow a : \{F(N^1_b)\}K^1_{ab}$	D92, D90, modus-ponens
D94	$\Gamma\ 1.6\,b \lhd \{K^1_{ab}.a\}K_{bs} \Leftarrow$	D59, substituição
	$1.6\,b \leftarrow a : \{K^1_{ab}.a\}K_{bs}$	
D95	$\Gamma\ 1.6\,b \lhd \{K^1_{ab}.a\}K_{bs}$	D45, D94, modus-ponens
D96	$\Gamma\ 1.6\,b \lhd K^1_{ab}.a \Leftarrow$	D72, substituição
	$\Gamma\ 1.6\,b \lhd \{K^1_{ab}.a\}K_{bs}$,	
	$\Gamma\ 1.6\,b \lhd K_{bs}$	
D97	$\Gamma\ 1.6\,b \lhd K_{bs}$	D46, substituição
D98	$\Gamma\ 1.6\,b \lhd \{K^1_{ab}.a\}K_{bs},\ 1.6\,b \lhd K_{bs}$	D95, D97, conjunção
D99	$\Gamma\ 1.6\,b \lhd K^1_{ab}.a$	D96, D98, modus-ponens

D100	$\Gamma\ 1.6\, b \lhd K^1_{ab}.a \Leftrightarrow$ $1.6\, b \lhd K^1_{ab}$, $1.6\, b \lhd a$	D75, substituição
D101	$\Gamma\ 1.6\, b \lhd K^1_{ab},\ 1.6\, b \lhd a \Leftarrow$ $1.6\, b \lhd K^1_{ab}.a$	D100, bi-implicação
D102	$\Gamma\ 1.6\, b \lhd K^1_{ab},\ 1.6\, b \lhd a$	D99, D101, modus-ponens
D103	$\Gamma\ 1.6\, b \lhd K^1_{ab}$	D102, conjunção
D104	$\Gamma\ 1.11\, b \lhd K^1_{ab} \Leftarrow$ $1.6\, b \lhd K^1_{ab}$, $6 < 11$	D62, substituição
D105	$\Gamma\ 6 < 11$	D8, substituição
D106	$\Gamma\ 1.6\, b \lhd K^1_{ab},\ 6 < 11$	D103, D105, conjunção
D107	$\Gamma\ 1.11\, b \lhd K^1_{ab}$	D106, D104, modus-ponens
D108	$\Gamma\ 1.10\, b \leftarrow a : \{F(N^1_b)\}K^1_{ab},\ 1.11\, b \lhd K^1_{ab}$	D93, D107, conjunção
D109	$\Gamma\ 1.7\, b \boxtimes N^1_b$, $1.7\, b \overset{F(N^1_b)}{\sim} a$	D53, conjunção
D110	$\Gamma\ 1.10\, b \leftarrow a : \{F(N^1_b)\}K^1_{ab}$, $1.11\, b \lhd K^1_{ab}$, $1.7\, b \boxtimes N^1_b$, $1.7\, b \overset{F(N^1_b)}{\sim} a$	D108, D109, conjunção
D111	$\Gamma\ 7 < 10$	D8, substituição
D112	$\Gamma\ 10 < 11$	D8, substituição
D113	$\Gamma\ 10 < 11, 7 < 10$	D112, D111, conjunção
D114	$\Gamma\ 1.10\, b \leftarrow a : \{F(N^1_b)\}K^1_{ab}$, $1.11\, b \lhd K^1_{ab}$, $1.7\, b \boxtimes F(N^1_b)$, $1.7\, b \overset{F(N^1_b)}{\sim} a$, $10 < 11$, $7 < 10$	D110, D113, conjunção
D115	Γ P7	suposição
D116	$\Gamma\ 1.11\, b \overset{K^1_{ab}}{\leftrightarrow} a \Leftarrow$ $1.10\, b \leftarrow a : \{F(N^1_b)\}K^1_{ab}$, $1.11\, b \lhd K^1_{ab}$, $1.7\, b \boxtimes F(N^1_b)$, $1.7\, b \overset{F(N^1_b)}{\sim} a$, $10 < 11$, $7 < 10$	D115, substituição

D117	$\Gamma\ 1.11\, b \overset{K^1_{ab}}{\leftrightarrow} a$	D114, D116, modus-ponens
D118	$\Gamma\ 1.5\, a \overset{K^1_{ab}}{\leftrightarrow} b$	D40, conjunção
D119	$\Gamma\ 1.5\, a \overset{K^1_{ab}}{\leftrightarrow} b,\ 1.11\, b \overset{K^1_{ab}}{\leftrightarrow} a$	D118, D117, conjunção
D120	$\Gamma\ \exists\ E, I_1, I_2$ $E.I_1\, a \overset{K^1_{ab}}{\leftrightarrow} b,$ $E.I_2\, b \overset{K^1_{ab}}{\leftrightarrow} a$	D119, existencial

□

Teorema 5.3, página 106:

$$\Phi_{ns} \cup \Delta_{ep} \vdash\ \exists\ E_1, E_2, I_1, I_2\ E_1.I_1\, b \overset{K^{E_1}_{ab}}{\leftrightarrow} a,\ E_2.I_2\, b \overset{K^{E_1}_{ab}}{\leftrightarrow} a,\ E_2 < E_1$$

Demonstração.

D121	Γ CV	suposição
D122	$\Gamma\ 1.6\, c \leftarrow a : \{K^1_{ab}.a\}K_{bs} \Leftarrow$ $1.5\, a \rightarrow b : \{K^1_{ab}.a\}K_{bs},$ $5 < 6$	D121, substituição
D123	$\Gamma\ 1.5\, a \rightarrow b : \{K^1_{ab}.a\}K_{bs},\ 5 < 6$	D41, D43, conjunção
D124	$\Gamma\ 1.6\, c \leftarrow a : \{K^1_{ab}.a\}K_{bs}$	D123, D122, modus-ponens
D125	$\Gamma\ 1.6\, c \lhd \{K^1_{ab}.a\}K_{bs} \Leftarrow$ $1.6\, c \leftarrow a : \{K^1_{ab}.a\}K_{bs}$	D59, substituição
D126	$\Gamma\ 1.6\, c \lhd \{K^1_{ab}.a\}K_{bs}$	D124, D125, modus-ponens
D127	Γ PEP	suposição
D128	$\Gamma\ 2.1\, c \lhd \{K^1_{ab}.a\}K_{bs} \Leftarrow$ $1.6\, c \lhd \{K^1_{ab}.a\}K_{bs},$ $1 < 2$	D127, substituição
D129	$\Gamma\ 1.6\, c \lhd \{K^1_{ab}.a\}K_{bs},\ 1 < 2$	D126, D9, conjunção
D130	$\Gamma\ 2.1\, c \lhd \{K^1_{ab}.a\}K_{bs}$	D129, D128, modus-ponens
D131	Γ PER	suposição
D132	$\Gamma\ 2.1\, c_a \rightarrow b : \{K^1_{ab}.a\}K_{bs} \Leftarrow$ $2.1\, c \lhd \{K^1_{ab}.a\}K_{bs}$	D131, substituição
D133	$\Gamma\ 2.1\, c_a \rightarrow b : \{K^1_{ab}.a\}K_{bs}$	D130, D132, modus-ponens

D134	$\Gamma\ 2.2\, b \leftarrow c_a : \{K^1_{ab}.a\}K_{bs} \Leftarrow$ $2.1\, c_a \rightarrow b : \{K^1_{ab}.a\}K_{bs},$ $1 < 2$	D6, substituição
D135	$\Gamma\ 2.1\, c_a \rightarrow b : \{K^1_{ab}.a\}K_{bs},\ 1 < 2$	D133, D9, conjunção
D136	$\Gamma\ 2.2\, b \leftarrow c_a : \{K^1_{ab}.a\}K_{bs}$	D135, D134, modus-ponens
D137	Γ RIT	suposição
D138	$\Gamma\ 2.2\, b \leftarrow a : \{K^1_{ab}.a\}K_{bs} \Leftarrow$ $\Gamma\ 2.2\, b \leftarrow c_a : \{K^1_{ab}.a\}K_{bs}$	D137, substituição
D139	$\Gamma\ 2.2\, b \leftarrow a : \{K^1_{ab}.a\}K_{bs}$	D136, D138, modus-ponens
D140	$\Gamma\ 2.3\, b \rightarrow a : \{N^2_b\}K^1_{ab},$ $2.3\, b \boxtimes F(N^2_b),$ $2.3\, b \overset{F(N^2_b)}{\sim} a \Leftarrow$ $2.2\, b \leftarrow a : \{K^1_{ab}.a\}K_{bs}, 2.3\, b \lhd K_{bs}, 2 < 3$	D51, substituição
D141	$\Gamma\ 2.3\, b \lhd K_{bs}$	D46, substituição
D142	$\Gamma\ 2.2\, b \leftarrow c_a : \{K^1_{ab}.a\}K_{bs},$ $2.3\, b \lhd K_{bs}$	D139, D141, conjunção
D143	$\Gamma\ 2.2\, b \leftarrow c_a : \{K^1_{ab}.a\}K_{bs},$ $2.3\, b \lhd K_{bs},$ $2 < 3$	D142, D18, conjunção
D144	$\Gamma\ 2.3\, b \rightarrow a : \{N^2_b\}K^1_{ab},$ $2.3\, b \boxtimes F(N^2_b),$ $2.3\, b \overset{F(N^2_b)}{\sim} a$	D143, D140, modus-ponens
D145	$\Gamma\ 2.3\, b \rightarrow a : \{N^2_b\}K^1_{ab}$	D144, conjunção
D146	$\Gamma\ 2.4\, c \leftarrow b : \{N^2_b\}K^1_{ab} \Leftarrow$ $2.3\, b \rightarrow a : \{N^2_b\}K^1_{ab},$ $3 < 4$	D121, substituição
D147	$\Gamma\ 2.3\, b \rightarrow a : \{N^2_b\}K^1_{ab},\ 3 < 4$	D145, D27, conjunção
D148	$\Gamma\ 2.4\, c \leftarrow b : \{N^2_b\}K^1_{ab}$	D147, D146, modus-ponens
D149	$\Gamma\ 2.4\, c \lhd \{N^2_b\}K^1_{ab} \Leftarrow$ $2.4\, c \leftarrow b : \{N^2_b\}K^1_{ab}$	D59, substituição
D150	$\Gamma\ 2.4\, c \lhd \{N^2_b\}K^1_{ab}$	D148, D149, modus-ponens
D151	Γ TI	suposição
D152	$\Gamma\ 2.4\, c \lhd K^1_{ab} \Leftarrow$ $1.8\, c \lhd \{N^1_b\}K^1_{ab},$ $1 < 2$	D151, substituição
D153	$\Gamma\ 1.8\, c \leftarrow b : \{N^1_b\}K^1_{ab} \Leftarrow$ $1.7\, b \rightarrow a : \{N^1_b\}K^1_{ab},$	D121, substituição

	$7 < 8$	
D154	$\Gamma\ 1.7\, b \rightarrow a : \{N_b^1\}K_{ab}^1, 7 < 8$	D54, D56, conjunção
D155	$\Gamma\ 1.8\, c \leftarrow b : \{N_b^1\}K_{ab}^1$	D154, D153, modus-ponens
D156	$\Gamma\ 1.8\, c \lhd \{N_b^1\}K_{ab}^1 \Leftarrow$	D59, substituição
	$1.8\, c \leftarrow b : \{N_b^1\}K_{ab}^1$	
D157	$\Gamma\ 1.8\, c \lhd \{N_b^1\}K_{ab}^1$	D155, D156, modus-ponens
D158	$\Gamma\ 1.8\, c \lhd \{N_b^1\}K_{ab}^1,\ 1 < 2$	D157, D9, conjunção
D159	$\Gamma\ 2.4\, c \lhd K_{ab}^1$	D158, D152, modus-ponens
D160	$\Gamma\ 2.4\, c \lhd \{N_b^2\}K_{ab}^1,\ 2.4\, c \lhd K_{ab}^1$	D150, D159, conjunção
D161	$\Gamma\ 2.4\, c \lhd N_b^2 \Leftarrow$	D72, substituição
	$2.4\, c \lhd \{N_b^2\}K_{ab}^1,$	
	$2.4\, c \lhd K_{ab}^1$	
D162	$\Gamma\ 2.4\, c \lhd N_b^2$	D160, D161, modus-ponens
D163	Γ PMM	suposição
D164	$\Gamma\ 2.4\, c \lhd F(N_b^2) \Leftarrow$	D163, substituição
	$2.4\, c \lhd N_b^2$	
D165	$\Gamma\ 2.4\, c \lhd F(N_b^2)$	D163, D164, modus-ponens
D166	Γ PMC	suposição
D167	$\Gamma\ 2.4\, c \lhd \{F(N_b^2)\}K_{ab}^1 \Leftarrow$	D166, substituição
	$2.4\, c \lhd F(N_b^2),$	
	$2.4\, c \lhd K_{ab}^1$	
D168	$\Gamma\ 2.4\, c \lhd F(N_b^2),\ 2.4\, c \lhd K_{ab}^1$	D165, D159, conjunção
D169	$\Gamma\ 2.4\, c \lhd \{F(N_b^2)\}K_{ab}^1$	D168, D167, modus-ponens
D170	$\Gamma\ 2.4\, c_a \rightarrow b : \{F(N_b^2)\}K_{ab}^1 \Leftarrow$	D131, substituição
	$2.4\, c \lhd \{F(N_b^2)\}K_{ab}^1$	
D171	$\Gamma\ 2.4\, c_a \rightarrow b : \{F(N_b^2)\}K_{ab}^1$	D169, D170, modus-ponens
D172	$\Gamma\ 2.5\, b \leftarrow c_a : \{F(N_b^2)\}K_{ab}^1 \Leftarrow$	D6, substituição
	$2.4\, c_a \rightarrow b : \{F(N_b)\}K_{ab}^2,$	
	$4 < 5$	
D173	$\Gamma\ 2.4\, c_a \rightarrow b : \{F(N_b^2)\}K_{ab}^1,\ 4 < 5$	D171, D33, conjunção
D174	$\Gamma\ 2.5\, b \leftarrow c_a : \{F(N_b^2)\}K_{ab}^1$	D173, D172, modus-ponens
D175	$\Gamma\ 2.5\, b \leftarrow a : \{F(N_b^2)\}K_{ab}^1 \Leftarrow$	D137, substituição
	$2.5\, b \leftarrow c_a : \{F(N_b^2)\}K_{ab}^1$	
D176	$\Gamma\ 2.5\, b \leftarrow a : \{F(N_b^2)\}K_{ab}^1$	D174, D175, modus-ponens
D177	$\Gamma\ 2.2\, b \lhd K_{ab}^1.a \Leftarrow$	D72, substituição
	$2.2\, b \lhd \{K_{ab}^1.a\}K_{bs},$	
	$2.2\, b \lhd K_{bs}$	
D178	$\Gamma\ 2.2\, b \lhd \{K_{ab}^1.a\}K_{bs} \Leftarrow$	D59, substituição

$2.2\,b \leftarrow a : \{K^1_{ab}.a\}K_{bs}$

D179 $\Gamma\ 2.2\,b \lhd \{K^1_{ab}.a\}K_{bs}$ — D139, D178, modus-ponens

D180 $\Gamma\ 2.2\,b \lhd K_{bs}$ — D46, substituição

D181 $\Gamma\ 2.2\,b \lhd \{K^1_{ab}.a\}K_{bs},\ 2.2\,b \lhd K_{bs}$ — D179, D180, conjunção

D182 $\Gamma\ 2.2\,b \lhd K^1_{ab}.a$ — D181, D177, modus-ponens

D183 $\Gamma\ 2.2\,b \lhd K^1_{ab}.a \Leftrightarrow$ — D75, substituição
$2.2\,b \lhd K^1_{ab}$,
$2.2\,b \lhd a$

D184 $\Gamma\ 2.2\,b \lhd K^1_{ab},\ 2.2\,b \lhd a \Leftarrow$ — D183, bi-implicação
$2.2\,b \lhd K^1_{ab}.a$

D185 $\Gamma\ 2.2\,b \lhd K^1_{ab},\ 2.2\,b \lhd a$ — D182, D184, modus-ponens

D186 $\Gamma\ 2.2\,b \lhd K^1_{ab}$ — D185, conjunção

D187 $\Gamma\ 2.6\,b \lhd K^1_{ab} \Leftarrow$ — D62, substituição
$2.2\,b \lhd K^1_{ab}$,
$2 < 6$

D188 $\Gamma\ 2 < 6$ — D8, substituição

D189 $\Gamma\ 2.2\,b \lhd K^1_{ab},\ 2 < 6$ — D186, D188, conjunção

D190 $\Gamma\ 2.6\,b \lhd K^1_{ab}$ — D189, D187, modus-ponens

D191 $\Gamma\ 2.5\,b \leftarrow a : \{F(N^2_b)\}K^1_{ab},\ 2.6\,b \lhd K^1_{ab}$ — D176, D190, conjunção

D192 $\Gamma\ 2.3\,b \boxtimes F(N^2_b)$, — D144, conjunção
$2.3\,b \overset{F(N^2_b)}{\sim} a$

D193 $\Gamma\ 2.5\,b \leftarrow a : \{F(N^2_b)\}K^1_{ab}$, — D191, D192, conjunção
$2.6\,b \lhd K^1_{ab}$,
$2.3\,b \boxtimes F(N^2_b)$,
$2.3\,b \overset{F(N^2_b)}{\sim} a$

D194 $\Gamma\ 3 < 5$ — D8, substituição

D195 $\Gamma\ 5 < 6,\ 3 < 5$ — D43, D194, conjunção

D196 $\Gamma\ 2.5\,b \leftarrow a : \{F(N^2_b)\}K^1_{ab}$, — D193, D195, conjunção
$2.6\,b \lhd K^1_{ab}$,
$2.3\,b \boxtimes F(N^2_b)$,
$2.3\,b \overset{F(N^2_b)}{\sim} a$,
$5 < 6$,
$3 < 5$

D197 $\Gamma\ 2.6\,b \overset{K^1_{ab}}{\leftrightarrow} a \Leftarrow$ — D115, substituição
$2.5\,b \leftarrow a : \{F(N^2_b)\}K^1_{ab}$,
$2.6\,b \lhd K^1_{ab}$,
$2.3\,b \boxtimes F(N^2_b)$,

		$2.3\, b \overset{F(N_b^2)}{\sim} a$,	
		$5 < 6$,	
		$3 < 5$	
D198	Γ	$2.6\, b \overset{K_{ab}^1}{\leftrightarrow} a$	D196, D197, modus-ponens
D199	Γ	$2.6\, b \overset{K_{ab}^1}{\leftrightarrow} a,\ 1.11\, b \overset{K_{ab}^1}{\leftrightarrow} a$	D198, D117, conjunção
D200	Γ	$2.6\, b \overset{K_{ab}^1}{\leftrightarrow} a$,	D199, D9, conjunção
		$1.11\, b \overset{K_{ab}^1}{\leftrightarrow} a$,	
		$1 < 2$	
D201	Γ	$\exists\ E_1, E_2, I_1, I_2$	D200, existencial
		$E_1.I_1\, b \overset{K_{ab}^{E_1}}{\leftrightarrow} a$,	
		$E_2.I_2\, b \overset{K_{ab}^{E_1}}{\leftrightarrow} a$,	
		$E_2 < E_1$	

□

Teorema 5.4, página 109:

$$\Phi'_{ns} \cup \Delta_{ep} \vdash \ \exists\ E_1, E_2, I_1, I_2\ E_1.I_1\, a \overset{K_{ab}^{E_1}}{\leftrightarrow} b,\ E_2.I_2\, a \overset{K_{ab}^{E_1}}{\leftrightarrow} b,\ E_1 < E_2$$

Demonstração.

D121	Γ	$2.1\, a \overset{N_a^2}{\sim} b$	D1, substituição
D122	Γ	$2.1\, a \to b : a.b.N_a^2 \Leftarrow$	D3, substituição
		$2.1\, a \overset{N_a^2}{\sim} b$	
D123	Γ	$2.1\, a \to b : a.b.N_a^2$	D121, D122, modus-ponens
D124	Γ	CV	suposição
D125	Γ	$1.4\, c \leftarrow s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as} \Leftarrow$	D124, substituição
		$1.3\, s \to a : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$,	
		$3 < 4$	
D126	Γ	$1.4\, c \leftarrow s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$	D28, D125, modus-ponens
D127	Γ	PR	suposição
D128	Γ	$1.4\, c \lhd \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as} \Leftarrow$	D127, substituição
		$1.4\, c \leftarrow s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$	

D129 Γ $1.4\, c \lhd \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$ — D126, D128, modus-ponens

D130 Γ PEP — suposição

D131 Γ $2.2\, c \lhd \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as} \Leftarrow$ — D130, substituição
$1.4\, c \lhd \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$,
$1 < 2$

D132 Γ $1.4\, c \lhd \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}, 1 < 2$ — D129, D9, conjunção

D133 Γ $2.2\, c \lhd \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$ — D132, D131, modus-ponens

D134 Γ PER

D135 Γ $2.2\, c_s \rightarrow a : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as} \Leftarrow$ — D134, substituição
$2.2\, c \lhd \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$

D136 Γ $2.2\, c_s \rightarrow a : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$ — D133, D135, modus-ponens

D137 Γ $2.3\, a \leftarrow c_s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as} \Leftarrow$ — D6, substituição
$2.2\, c_s \rightarrow a : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$,
$2 < 3$

D138 Γ $2.2\, c_s \rightarrow a : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}, 2 < 3$ — D136, D18, conjunção

D139 Γ $2.3\, a \leftarrow c_s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$ — D138, D137, modus-ponens

D140 Γ RIT

D141 Γ $2.3\, a \leftarrow s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as} \Leftarrow$ — D140, substituição
$2.3\, a \leftarrow c_s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$

D142 Γ $2.3\, a \leftarrow s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$ — D139, D141, modus-ponens

D143 Γ $2.4\, a \rightarrow b : \{K_{ab}^1.a\}K_{bs}$, — D38, substituição
$2.4\, a \overset{K_{ab}^1}{\leftrightarrow} b \Leftarrow$
$2.3\, a \leftarrow s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$,
$2.4\, a \lhd K_{as}$,
$2.1\, a \overset{N_a^2}{\sim} b$,
$3 < 4$,
$1 < 3$

D144 Γ $2.4\, a \lhd K_{as}$ — D30, substituição

D145 Γ $1 < 3$ — D8, substituição

D146 Γ $2.3\, a \leftarrow s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$, — D142, D144, conjunção
$2.4\, a \lhd K_{as}$

D147 Γ $2.3\, a \leftarrow s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$, — D146, D121, conjunção
$2.4\, a \lhd K_{as}$,
$2.1\, a \overset{N_a^2}{\sim} b$

D148 Γ $2.3\, a \leftarrow s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$, — D147, D27, conjunção
$2.4\, a \lhd K_{as}$,
$2.1\, a \overset{N_a^2}{\sim} b$,

$3 < 4$

D149 Γ $2.3\, a \leftarrow s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$, D148, D145, conjunção
$2.4\, a \lhd K_{as}$,
$2.1\, a \overset{N_a^2}{\sim} b$,
$3 < 4$,
$1 < 3$

D150 Γ $2.4\, a \rightarrow b : \{K_{ab}^1.a\}K_{bs}$, D149, D143, modus-ponens
$2.4\, a \overset{K_{ab}^1}{\leftrightarrow} b$

D151 Γ $2.4\, a \overset{K_{ab}^1}{\leftrightarrow} b$ D150, conjunção

D152 Γ $1.5\, a \overset{K_{ab}^1}{\leftrightarrow} b$ D40, conjunção

D153 Γ $1.5\, a \overset{K_{ab}^1}{\leftrightarrow} b, 2.4\, a \overset{K_{ab}^1}{\leftrightarrow} b$ D152, D151, conjunção

D154 Γ $1.5\, a \overset{K_{ab}^1}{\leftrightarrow} b, 2.4\, a \overset{K_{ab}^1}{\leftrightarrow} b, 1 < 2$ D153, D9, conjunção

D155 Γ $\exists\, E_1, E_2, I_1, I_2$, D154, existencial
$E_1.I_1\, a \overset{K_{ab}^{E_1}}{\leftrightarrow} b$,
$E_2.I_2\, a \overset{K_{ab}^{E_1}}{\leftrightarrow} b$,
$E_1 < E_2$

□

Continuação do cenário de ataque do protocolo 5.4, página 109:

D156 Γ $1.8\, c \leftarrow b : \{N_b^1\}K_{ab}^1 \Leftarrow$ D124, substituição
$1.7\, b \rightarrow a : \{N_b^1\}K_{ab}^1$,
$7 < 8$

D157 Γ $1.8\, c \leftarrow b : \{N_b^1\}K_{ab}^1$ D57, D156, modus-ponens

D158 Γ $1.8\, c \lhd \{N_b^1\}K_{ab}^1 \Leftarrow$ D59, substituição
$1.8\, c \leftarrow b : \{N_b^1\}K_{ab}^1$

D159 Γ $1.8\, c \lhd \{N_b^1\}K_{ab}^1$ D157, D158, modus-ponens

D160 Γ $2.5\, c \lhd \{N_b^1\}K_{ab}^1 \Leftarrow$ D130, substituição
$1.8\, c \lhd \{N_b^1\}K_{ab}^1$,
$1 < 2$

D161 Γ $1.8\, c \lhd \{N_b^1\}K_{ab}^1$, D159, D9, conjunção
$1 < 2$

D162 Γ $2.5\, c \lhd \{N_b^1\}K_{ab}^1$ D161, D160, modus-ponens

D163 Γ $2.5\, c_b \rightarrow a : \{N_b^1\}K_{ab}^1 \Leftarrow$ D134, substituição
$2.5\, c \lhd \{N_b^1\}K_{ab}^1$

D164 Γ $2.5\, c_b \rightarrow a : \{N_b^1\}K_{ab}^1$ D162, D163, modus-ponens

D165 Γ $2.6\, a \leftarrow c_b : \{N_b^1\}K_{ab}^1 \Leftarrow$ — D6, substituição
$2.5\, c_b \rightarrow a : \{N_b^1\}K_{ab}^1$,
$5 < 6$

D166 Γ $2.5\, c_b \rightarrow a : \{N_b^1\}K_{ab}^1$, — D164, D43, conjunção
$5 < 6$

D167 Γ $2.6\, a \leftarrow c_b : \{N_b^1\}K_{ab}^1$ — D166, D165, modus-ponens

D168 Γ $2.6\, a \leftarrow b : \{N_b^1\}K_{ab}^1 \Leftarrow$ — D140, substituição
$2.6\, a \leftarrow c_b : \{N_b^1\}K_{ab}^1$

D169 Γ $2.6\, a \leftarrow b : \{N_b^1\}K_{ab}^1$ — D167, D168, modus-ponens

D170 Γ $2.7\, a \rightarrow b : \{F(N_b^1)\}K_{ab}^1 \Leftarrow$ — D87, substituição
$2.6\, a \leftarrow b : \{N_b^1\}K_{ab}^1$,
$2.6\, a \lhd K_{ab}^1$,
$6 < 7$

D171 Γ $2.3\, a \lhd \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as} \Leftarrow$ — D59, substituição
$2.3\, a \leftarrow s : \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$

D172 Γ $2.3\, a \lhd \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$ — D142, D171, modus-ponens

D173 Γ $2.3\, a \lhd N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs} \Leftarrow$ — D72, substituição
$2.3\, a \lhd \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$,
$2.3\, a \lhd K_{as}$

D174 Γ $2.3\, a \lhd K_{as}$ — D30, substituição

D175 Γ $2.3\, a \lhd \{N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}\}K_{as}$, — D172, D174, conjunção
$2.3\, a \lhd K_{as}$

D176 Γ $2.3\, a \lhd N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}$ — D175, D173, modus-ponens

D177 Γ $2.3\, a \lhd N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs} \Leftrightarrow$ — D75, substituição
$2.3\, a \lhd N_a^1.b.K_{ab}^1, 2.3\, a \lhd \{K_{ab}^1.a\}K_{bs}$

D178 Γ $2.3\, a \lhd N_a^1.b.K_{ab}^1, 2.3\, a \lhd \{K_{ab}^1.a\}K_{bs} \Leftarrow$ — D177, bi-implicação
$2.3\, a \lhd N_a^1.b.K_{ab}^1\{K_{ab}^1.a\}K_{bs}$

D179 Γ $2.3\, a \lhd N_a^1.b.K_{ab}^1, 2.3\, a \lhd \{K_{ab}^1.a\}K_{bs}$ — D176, D178, modus-ponens

D180 Γ $2.3\, a \lhd N_a^1.b.K_{ab}^1$ — D179, conjunção

D181 Γ $2.3\, a \lhd N_a^1.b.K_{ab}^1 \Leftrightarrow$ — D75, substituição
$2.3\, a \lhd N_a^1.b, 2.3\, a \lhd K_{ab}^1$

D182 Γ $2.3\, a \lhd N_a^1.b, 2.3\, a \lhd K_{ab}^1 \Leftarrow$ — D181, bi-implicação
$2.3\, a \lhd N_a^1.b.K_{ab}^1$

D183 Γ $2.3\, a \lhd N_a^1.b, 2.3\, a \lhd K_{ab}^1$ — D180, D182, modus-ponens

D184 Γ $2.3\, a \lhd K_{ab}^1$ — D183, conjunção

D185 Γ $2.6\, a \lhd K_{ab}^1 \Leftarrow$ — D62, substituição
$2.3\, a \lhd K_{ab}^1$,
$3 < 6$

D186	$\Gamma\ 3 < 6$	D8, substituição
D187	$\Gamma\ 2.3\, a \lhd K^1_{ab}$, $3 < 6$	D184, D186, conjunção
D188	$\Gamma\ 2.6\, a \lhd K^1_{ab}$	D187, D185, modus-ponens
D189	$\Gamma\ 2.6\, a \leftarrow b : \{N^1_b\}K^1_{ab}$, $2.6\, a \lhd K^1_{ab}$	D169, D188, conjunção
D190	$\Gamma\ 2.6\, a \leftarrow b : \{N^1_b\}K^1_{ab}$, $2.6\, a \lhd K^1_{ab}$, $6 < 7$	D189, D49, conjunção
D191	$\Gamma\ 2.7\, a \rightarrow b : \{F(N^1_b)\}K^1_{ab}$	D190, D170, modus-ponens

A.2 Teoremas sobre o Protocolo Otway-Rees

Teorema 5.5, página 112:

$$\begin{aligned}\Phi_{or} \cup \Delta_{ep} \vdash\ & \exists\ E, I_1, I_2, CH \\ & E.I_1\, a \overset{CH}{\leftrightarrow} b, \\ & CH \neq K^E_{ab}, \\ & E.I_2\, c \lhd CH\end{aligned}$$

Demonstração.

D1	Γ P1	suposição
D2	$\Gamma\ 1.1\, a \rightarrow b : \overset{1}{N^1_a}.a.b.\{\overset{2}{N^1_a}.\overset{1}{N^1_a}.a.b\}K_{as}$, $1.1\, a \boxtimes \overset{1}{N^1_a}$, $1.1\, a \boxtimes \overset{2}{N^1_a}$, $1.1\, a \overset{\overset{2}{N^1_a}}{\sim} b$	D1, substituição
D3	$\Gamma\ 1.1\, a \rightarrow b : \overset{1}{N^1_a}.a.b.\{\overset{2}{N^1_a}.\overset{1}{N^1_a}.a.b\}K_{as}$	D2, conjunção
D4	$\Gamma\ 1.1\, a \boxtimes \overset{1}{N^1_a}$, $1.1\, a \boxtimes \overset{2}{N^1_a}$, $1.1\, a \overset{\overset{2}{N^1_a}}{\sim} b$	D2, conjunção
D5	$\Gamma\ 1.1\, a \boxtimes \overset{1}{N^1_a}$	D4, conjunção
D6	$\Gamma\ 1.1\, a \boxtimes \overset{2}{N^1_a}$, $1.1\, a \overset{\overset{2}{N^1_a}}{\sim} b$	D4, conjunção
D7	$\Gamma\ 1.1\, a \boxtimes \overset{2}{N^1_a}$	D6, conjunção

D8	$\Gamma\ 1.1\, a \overset{2}{\underset{N_a^1}{\sim}} b$	D6, conjunção
D9	Γ CV	suposição
D10	$\Gamma\ 1.2\, c \leftarrow a : \overset{1}{N_a^1}.a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as} \Leftarrow$ $1.2\, a \rightarrow b : \overset{1}{N_a^1}.a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as},$ $1 < 2$	D9, substituição
D11	Γ MQ	suposição
D12	$\Gamma\ 1 < 2$	D11, substituição
D13	$\Gamma\ 1.1\, a \rightarrow b : \overset{1}{N_a^1}.a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}, 1 < 2$	D3, D12, conjunção
D14	$\Gamma\ 1.2\, c \leftarrow a : \overset{1}{N_a^1}.a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D13, D10, modus-ponens
D15	Γ PR	
D16	$\Gamma\ 1.2\, c \lhd \overset{1}{N_a^1}.a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as} \Leftarrow$ $1.2\, c \leftarrow a : \overset{1}{N_a^1}.a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D15, substituição
D17	$\Gamma\ 1.2\, c \lhd \overset{1}{N_a^1}.a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D14, D16, modus-ponens
D18	Γ MS	suposição
D19	$\Gamma\ 1.2\, c \lhd \overset{1}{N_a^1}.a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as} \Leftrightarrow$ $1.2\, c \lhd \overset{1}{N_a^1}, 1.2\, c \lhd a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D18, substituição
D20	$\Gamma\ 1.2\, c \lhd \overset{1}{N_a^1}, 1.2\, c \lhd a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as} \Leftarrow$ $1.2\, c \lhd \overset{1}{N_a^1}.a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D19, bi-implicação
D21	$\Gamma\ 1.2\, c \lhd \overset{1}{N_a^1}, 1.2\, c \lhd a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D17, D20, modus-ponens
D22	$\Gamma\ 1.2\, c \lhd \overset{1}{N_a^1}$	D21, conjunção
D23	$\Gamma\ 1.2\, c \lhd a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D21, conjunção
D24	$\Gamma\ 1.2\, c \lhd a.b, 1.2\, c \lhd \{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as} \Leftrightarrow$ $1.2\, c \lhd a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D18, substituição
D25	$\Gamma\ 1.2\, c \lhd a.b, 1.2\, c \lhd \{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as} \Leftarrow$ $1.2\, c \lhd a.b.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D24, bi-implicação
D26	$\Gamma\ 1.2\, c \lhd a.b, 1.2\, c \lhd \{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D23, D25, modus-ponens
D27	$\Gamma\ 1.2\, c \lhd \{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D26, conjunção
D28	$\Gamma\ 1.2\, c \lhd \overset{1}{N_a^1}.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as} \Leftrightarrow$ $1.2\, c \lhd \overset{1}{N_a^1}, 1.2\, c \lhd \{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D18, substituição
D29	$\Gamma\ 1.2\, c \lhd \overset{1}{N_a^1}.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as} \Leftarrow$ $1.2\, c \lhd \overset{1}{N_a^1}, 1.2\, c \lhd \{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D28, bi-implicação

D30	$\Gamma\ 1.2\, c \lhd \overset{1}{N}{}^1_a, 1.2\, c \lhd \{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}$	D22, D27, conjunção
D31	$\Gamma\ 1.2\, c \lhd \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}$	D30, D29, modus-ponens
D32	$\Gamma\ 1.2\, c \lhd \overset{1}{N}{}^1_a.a.b \Leftrightarrow$ $1.2\, c \lhd \overset{1}{N}{}^1_a, 1.2\, c \lhd a.b$	D18, substituição
D33	$\Gamma\ 1.2\, c \lhd \overset{1}{N}{}^1_a.a.b \Leftarrow$ $1.2\, c \lhd \overset{1}{N}{}^1_a, 1.2\, c \lhd a.b$	D32, bi-implicação
D34	$\Gamma\ 1.2\, c \lhd a.b$	D26, conjunção
D35	$\Gamma\ 1.2\, c \lhd \overset{1}{N}{}^1_a, 1.2\, c \lhd a.b$	D22, D34, conjunção
D36	$\Gamma\ 1.2\, c \lhd \overset{1}{N}{}^1_a.a.b$	D35, D33, modus-ponens
D37	Γ PER	suposição
D38	$\Gamma\ 1.2\, c_b \rightarrow a : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as} \Leftarrow$ $1.2\, c \lhd \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}$	D37, substituição
D39	$\Gamma\ 1.2\, c_b \rightarrow a : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}$	D31, D38, modus-ponens
D40	Γ TN	suposição
D41	$\Gamma\ 1.3\, a \leftarrow c_b : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as} \Leftarrow$ $1.2\, c_b \rightarrow a : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as},$ $2 < 3$	D40, substituição
D42	$\Gamma\ 2 < 3$	D11, substituição
D43	$\Gamma\ 1.2\, c_b \rightarrow a : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}, 2 < 3$	D39, D42, conjunção
D44	$\Gamma\ 1.3\, a \leftarrow c_b : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}$	D43, D41, modus-ponens
D45	Γ RIT	suposição
D46	$\Gamma\ 1.3\, a \leftarrow b : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as} \Leftarrow$ $1.3\, a \leftarrow c_b : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}$	D45, substituição
D47	$\Gamma\ 1.3\, a \leftarrow b : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}$	D44, D46, modus-ponens
D48	Γ SI1	suposição
D49	$\Gamma\ 1.4\, a \lhd K_{as}$	D48, substituição
D50	Γ P5	suposição
D51	$\Gamma\ 1.4\, a \overset{\overset{1}{N}{}^1_a.a.b}{\leftrightarrow} b \Leftarrow$ $1.3\, a \leftarrow b : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as},$ $1.4\, a \lhd K_{as},$ $1.1\, a \boxtimes \overset{1}{N}{}^1_a,$ $1.1\, a \boxtimes \overset{2}{N}{}^1_a,$	D50, substituição

	$1.1\, a \overset{2}{\underset{N_a^1}{\sim}} b$,	
	$3 < 4$,	
	$1 < 3$,	
D52	$\Gamma\ 1.3\, a \leftarrow b : \overset{1}{N_a^1}.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$,	D47, D49, conjunção
	$1.4\, a \lhd K_{as}$	
D53	$\Gamma\ 1.3\, a \leftarrow b : \overset{1}{N_a^1}.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$,	D52, D5, conjunção
	$1.4\, a \lhd K_{as}$,	
	$1.1\, a \boxtimes \overset{1}{N_a^1}$	
D54	$\Gamma\ 1.3\, a \leftarrow b : \overset{1}{N_a^1}.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$,	D53, D7, conjunção
	$1.4\, a \lhd K_{as}$,	
	$1.1\, a \boxtimes \overset{1}{N_a^1}$,	
	$1.1\, a \boxtimes \overset{2}{N_a^1}$	
D55	$\Gamma\ 1.3\, a \leftarrow b : \overset{1}{N_a^1}.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$,	D54, D8, conjunção
	$1.4\, a \lhd K_{as}$,	
	$1.1\, a \boxtimes \overset{1}{N_a^1}$,	
	$1.1\, a \boxtimes \overset{2}{N_a^1}$,	
	$1.1\, a \overset{2}{\underset{N_a^1}{\sim}} b$	
D56	$\Gamma\ 3 < 4$	D11, substituição
D57	$\Gamma\ 1 < 3$	D11, substituição
D58	$\Gamma\ 1.3\, a \leftarrow b : \overset{1}{N_a^1}.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$,	D55, D56, conjunção
	$1.4\, a \lhd K_{as}$,	
	$1.1\, a \boxtimes \overset{1}{N_a^1}$,	
	$1.1\, a \boxtimes \overset{2}{N_a^1}$,	
	$1.1\, a \overset{2}{\underset{N_a^1}{\sim}} b$,	
	$3 < 4$	
D59	$\Gamma\ 1.3\, a \leftarrow b : \overset{1}{N_a^1}.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$,	D58, D57, conjunção
	$1.4\, a \lhd K_{as}$,	
	$1.1\, a \boxtimes \overset{1}{N_a^1}$,	
	$1.1\, a \boxtimes \overset{2}{N_a^1}$,	
	$1.1\, a \overset{2}{\underset{N_a^1}{\sim}} b$,	
	$3 < 4$,	
	$1 < 3$	

D60	$\Gamma\ 1.4\, a \overset{N^1_a.a.b}{\leftrightarrow} b$	D59, D51, modus-ponens
D61	Γ MDF	suposição
D62	$\Gamma\ \overset{1}{N^1_a}.a.b \neq K^1_{ab} \Leftarrow$ $\overset{1}{N^1_a} \neq K^1_{ab}$	D61, substituição
D63	Γ DTM	suposição
D64	$\Gamma\ \overset{1}{N^1_a} \neq K^1_{ab}$	D63, substituição
D65	$\Gamma\ \overset{1}{N^1_a}.a.b \neq K^1_{ab}$	D64, D62, modus-ponens
D66	$\Gamma\ 1.4\, a \overset{N^1_a.a.b}{\leftrightarrow} b, \overset{1}{N^1_a}.a.b \neq K^1_{ab}$	D60, D65, conjunção
D67	$\Gamma\ 1.4\, a \overset{N^1_a.a.b}{\leftrightarrow} b,$ $\overset{1}{N^1_a}.a.b \neq K^1_{ab},$ $1.2\, c \lhd \overset{1}{N^1_a}.a.b$	D66, D36, conjunção
D68	$\Gamma\ \exists\ E, I_1, I_2, CH$ $E.I_1\, a \overset{CH}{\leftrightarrow} b,$ $CH \neq K^E_{ab},$ $E.I_2\, c \lhd CH$	D67, existencial

□

Teorema 5.6, página 114:

$$\Phi'_{or} \cup \Delta_{ep} \vdash \begin{array}{l} \exists\ E, I_1, I_2, CH \\ E.I_1\, a \overset{CH}{\leftrightarrow} b, \\ CH \neq K^E_{ab}, \\ E.I_2\, c \lhd CH \end{array}$$

Demonstração.

D1	Γ P1	suposição
D2	$\Gamma\ 1.1\, a \rightarrow b : \overset{1}{N^1_a}.a.b.\{\overset{2}{N^1_a}.\overset{1}{N^1_a}.a.b\}K_{as},$ $1.1\, a \boxtimes \overset{1}{N^1_a},$ $1.1\, a \boxtimes \overset{2}{N^1_a},$ $1.1\, a \overset{\overset{2}{N^1_a}}{\sim} b$	D1, substituição
D3	$\Gamma\ 1.1\, a \rightarrow b : \overset{1}{N^1_a}.a.b.\{\overset{2}{N^1_a}.\overset{1}{N^1_a}.a.b\}K_{as}$	D2, conjunção
D4	$\Gamma\ 1.1\, a \boxtimes \overset{1}{N^1_a},$	D2, conjunção

	$1.1\, a \boxtimes \overset{2}{N}{}^{1}_{a}$,	
	$1.1\, a \overset{\overset{2}{N}{}^{1}_{a}}{\sim} b$	
D5	$\Gamma\ 1.1\, a \boxtimes \overset{1}{N}{}^{1}_{a}$	D4, conjunção
D6	$\Gamma\ 1.1\, a \boxtimes \overset{2}{N}{}^{1}_{a}$,	D4, conjunção
	$1.1\, a \overset{\overset{2}{N}{}^{1}_{a}}{\sim} b$	
D7	$\Gamma\ 1.1\, a \boxtimes \overset{2}{N}{}^{1}_{a}$	D6, conjunção
D8	$\Gamma\ 1.1\, a \overset{\overset{2}{N}{}^{1}_{a}}{\sim} b$	D6, conjunção
D9	Γ CV	suposição
D10	$\Gamma\ 1.2\, c \leftarrow a : \overset{1}{N}{}^{1}_{a}.a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as} \Leftarrow$ $1.1\, a \rightarrow b : \overset{1}{N}{}^{1}_{a}.a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as}$, $1 < 2$	D9, substituição
D11	Γ MQ	suposição
D12	$\Gamma\ 1 < 2$	D11, substituição
D13	$\Gamma\ 1.1\, a \rightarrow b : \overset{1}{N}{}^{1}_{a}.a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as}, 1 < 2$	D3, D12, conjunção
D14	$\Gamma\ 1.2\, c \leftarrow a : \overset{1}{N}{}^{1}_{a}.a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as}$	D13, D10, modus-ponens
D15	Γ PR	
D16	$\Gamma\ 1.2\, c \lhd \overset{1}{N}{}^{1}_{a}.a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as} \Leftarrow$ $1.2\, c \leftarrow a : \overset{1}{N}{}^{1}_{a}.a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as}$	D15, substituição
D17	$\Gamma\ 1.2\, c \lhd \overset{1}{N}{}^{1}_{a}.a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as}$	D14, D16, modus-ponens
D18	Γ MS	suposição
D19	$\Gamma\ 1.2\, c \lhd \overset{1}{N}{}^{1}_{a}, 1.2\, c \lhd a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as} \Leftrightarrow$ $1.2\, c \lhd \overset{1}{N}{}^{1}_{a}.a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as}$	D18, substituição
D20	$\Gamma\ 1.2\, c \lhd \overset{1}{N}{}^{1}_{a}, 1.2\, c \lhd a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as} \Leftarrow$ $1.2\, c \lhd \overset{1}{N}{}^{1}_{a}.a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as}$	D19, bi-implicação
D21	$\Gamma\ 1.2\, c \lhd \overset{1}{N}{}^{1}_{a}, 1.2\, c \lhd a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as}$	D17, D20, modus-ponens
D22	$\Gamma\ 1.2\, c \lhd \overset{1}{N}{}^{1}_{a}$	D21, conjunção
D23	$\Gamma\ 1.2\, c \lhd a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as}$	D21, conjunção
D24	$\Gamma\ 1.2\, c \lhd a.b, 1.2\, c \lhd \{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as} \Leftrightarrow$ $1.2\, c \lhd a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as}$	D18, substituição
D25	$\Gamma\ 1.2\, c \lhd a.b, 1.2\, c \lhd \{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as} \Leftarrow$ $1.2\, c \lhd a.b.\{\overset{2}{N}{}^{1}_{a}.\overset{1}{N}{}^{1}_{a}.a.b\}K_{as}$	D24, bi-implicação

D26	$\Gamma\ 1.2\,c \lhd a.b, 1.2\,c \lhd \{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}$	D23, D25, modus-ponens
D27	$\Gamma\ 1.2\,c \lhd \{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}$	D26, conjunção
D28	Γ PID	suposição
D29	$\Gamma\ 1.2\,c \lhd a$	D28, substituição
D30	$\Gamma\ 1.2\,c \lhd b$	D28, substituição
D31	$\Gamma\ 1.2\,c \lhd \overset{1}{N}{}^1_a.a \Leftrightarrow$ $1.2\,c \lhd \overset{1}{N}{}^1_a,$ $1.2\,c \lhd a$	D18, substituição
D32	$\Gamma\ 1.2\,c \lhd \overset{1}{N}{}^1_a.a \Leftarrow$ $1.2\,c \lhd \overset{1}{N}{}^1_a,$ $1.2\,c \lhd a$	D31, bi-implicação
D33	$\Gamma\ 1.2\,c \lhd \overset{1}{N}{}^1_a,$ $1.2\,c \lhd a$	D22, D29, conjunção
D34	$\Gamma\ 1.2\,c \lhd \overset{1}{N}{}^1_a.a$	D33, D32, modus-ponens
D35	$\Gamma\ 1.2\,c \lhd \overset{1}{N}{}^1_a.a.c \Leftrightarrow$ $1.2\,c \lhd \overset{1}{N}{}^1_a.a,$ $1.2\,c \lhd c$	D18, substituição
D36	$\Gamma\ 1.2\,c \lhd \overset{1}{N}{}^1_a.a.c \Leftarrow$ $1.2\,c \lhd \overset{1}{N}{}^1_a.a,$ $1.2\,c \lhd c$	D35, bi-implicação
D37	$\Gamma\ 1.2\,c \lhd c$	D28, substituição
D38	$\Gamma\ 1.2\,c \lhd \overset{1}{N}{}^1_a.a, 1.2\,c \lhd c$	D34, D37, conjunção
D39	$\Gamma\ 1.2\,c \lhd \overset{1}{N}{}^1_a.a.c$	D38, D36, modus-ponens
D40	$\Gamma\ 1.2\,c \lhd \overset{1}{N}{}^1_a.a.b \Leftrightarrow$ $1.2\,c \lhd \overset{1}{N}{}^1_a.a,$ $1.2\,c \lhd b$	D18, substituição
D41	$\Gamma\ 1.2\,c \lhd \overset{1}{N}{}^1_a.a.b \Leftarrow$ $1.2\,c \lhd \overset{1}{N}{}^1_a.a,$ $1.2\,c \lhd b$	D40, bi-implicação
D42	$\Gamma\ 1.2\,c \lhd \overset{1}{N}{}^1_a.a,$ $1.2\,c \lhd b$	D34, D30, conjunção
D43	$\Gamma\ 1.2\,c \lhd \overset{1}{N}{}^1_a.a.b$	D42, D41, modus-ponens
D44	Γ PNC	suposição

D45	$\Gamma\ 1.2\, c \lhd N_c^1$	D44, substituição
D46	$\Gamma\ 1.2\, c \lhd N_c^1.\overset{1}{N_a^1}.a.b \Leftrightarrow$ $1.2\, c \lhd N_c^1,$ $1.2\, c \lhd \overset{1}{N_a^1}.a.b$	D18, substituição
D47	$\Gamma\ 1.2\, c \lhd N_c^1.\overset{1}{N_a^1}.a.b \Leftarrow$ $1.2\, c \lhd N_c^1,$ $1.2\, c \lhd \overset{1}{N_a^1}.a$	D46, bi-implicação
D48	$\Gamma\ 1.2\, c \lhd N_c^1,$ $1.2\, c \lhd \overset{1}{N_a^1}.a.b$	D45, D43, conjunção
D49	$\Gamma\ 1.2\, c \lhd N_c^1.\overset{1}{N_a^1}.a.b$	D48, D47, modus-ponens
D50	Γ SI6	suposição
D51	$\Gamma\ 1.2\, c \lhd K_{cs}$	D50, substituição
D52	Γ PMC	suposição
D53	$\Gamma\ 1.2\, c \lhd \{N_c^1.\overset{1}{N_a^1}.a.b\}K_{cs} \Leftarrow$ $1.2\, c \lhd N_c^1.\overset{1}{N_a^1}.a.b,$ $1.2\, c \lhd K_{cs}$	D52, substituição
D54	$\Gamma\ 1.2\, c \lhd N_c^1.\overset{1}{N_a^1}.a.b,$ $1.2\, c \lhd K_{cs}$	D49, D51, conjunção
D55	$\Gamma\ 1.2\, c \lhd \{N_c^1.\overset{1}{N_a^1}.a.b\}K_{cs}$	D54, D53, modus-ponens
D56	$\Gamma\ 1.2\, c \lhd \overset{1}{N_a^1}.a.c.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as} \Leftrightarrow$ $1.2\, c \lhd \overset{1}{N_a^1}.a.c,$ $1.2\, c \lhd \{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D18, substituição
D57	$\Gamma\ 1.2\, c \lhd \overset{1}{N_a^1}.a.c.\{N_c^1.\overset{1}{N_a^1}.a.b\}K_{as} \Leftarrow$ $1.2\, c \lhd \overset{1}{N_a^1}.a.c,$ $1.2\, c \lhd \{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D56, bi-implicação
D58	$\Gamma\ 1.2\, c \lhd \overset{1}{N_a^1}.a.c,$ $1.2\, c \lhd \{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D39, D27, conjunção
D59	$\Gamma\ 1.2\, c \lhd \overset{1}{N_a^1}.a.c.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}$	D58, D57, modus-ponens
D60	$\Gamma\ 1.2\, c \lhd \overset{1}{N_a^1}.a.c.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}.\{N_c^1.\overset{1}{N_a^1}.a.b\}K_{cs} \Leftrightarrow$ $1.2\, c \lhd \overset{1}{N_a^1}.a.c.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as},$ $1.2\, c \lhd \{N_c^1.\overset{1}{N_a^1}.a.b\}K_{cs}$	D18, substituição
D61	$\Gamma\ 1.2\, c \lhd \overset{1}{N_a^1}.a.c.\{\overset{2}{N_a^1}.\overset{1}{N_a^1}.a.b\}K_{as}.\{N_c^1.\overset{1}{N_a^1}.a.b\}K_{cs} \Leftarrow$	D60, bi-implicação

$1.2\, c \lhd \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}$,
$1.2\, c \lhd \{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs}$

D62 Γ $1.2\, c \lhd \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}$, $1.2\, c \lhd \{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs}$ — D59, D55, conjunção

D63 Γ $1.2\, c \lhd \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}.\{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs}$ — D62, D61, modus-ponens

D64 Γ PER

D65 Γ $1.2\, c_a \rightarrow s : \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}.\{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs} \Leftarrow$ $1.2\, c \lhd \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}.\{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs}$ — D64, substituição

D66 Γ $1.2\, c_a \rightarrow s : \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}.\{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs}$ — D63, D65, modus-ponens

D67 Γ TN

D68 Γ $1.3\, s \leftarrow c_a : \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}.\{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs} \Leftarrow$ $1.2\, c_a \rightarrow s : \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}.\{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs}$, $2 < 3$ — D67, substituição

D69 Γ $2 < 3$ — D11, substituição

D70 Γ $1.2\, c_a \rightarrow s : \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}.\{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs}$, $2 < 3$ — D66, D69, conjunção

D71 Γ $1.3\, s \leftarrow c_a : \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}.\{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs}$ — D70, D68, modus-ponens

D72 Γ RIT — suposição

D73 Γ $1.3\, s \leftarrow a : \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}.\{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs} \Leftarrow$ $1.3\, s \leftarrow c_a : \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}.\{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs}$ — D72, substituição

D74 Γ $1.3\, s \leftarrow a : \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}.\{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs} \Leftarrow$ — D71, D73, modus-ponens

D75 Γ SI2 — suposição

D76 Γ $1.4\, s \lhd K_{as}$ — D75, substituição

D77 Γ SI7 — suposição

D78 Γ $1.4\, s \lhd K_{cs}$ — D77, substituição

D79 Γ $3 < 4$ — D11, substituição

D80 Γ $1.3\, s \leftarrow a : \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}.\{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs}$, $1.4\, s \lhd K_{as}$ — D74, D76, conjunção

D81 Γ $1.3\, s \leftarrow a : \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}.\{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs}$, $1.4\, s \lhd K_{as}$, $1.4\, s \lhd K_{cs}$ — D80, D78, conjunção

D82 Γ $1.3\, s \leftarrow a : \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}.\{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs}$, $1.4\, s \lhd K_{as}$, — D81, D79, conjunção

	$1.4\, s \lhd K_{cs}$,	
	$3 < 4$	
D83	Γ P3	suposição
D84	$\Gamma\ 1.4\, s \rightarrow c : \overset{1}{N}{}^1_a\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}.\{N^1_c.K^1_{ac}\}K_{cs} \Leftarrow$ $1.3\, s \leftarrow a : \overset{1}{N}{}^1_a.a.c.\{\overset{2}{N}{}^1_a.\overset{1}{N}{}^1_a.a.b\}K_{as}.\{N^1_c.\overset{1}{N}{}^1_a.a.b\}K_{cs}$, $1.4\, s \lhd K_{as}$, $1.4\, s \lhd K_{cs}$, $3 < 4$	D83, substituição
D85	$\Gamma\ 1.4\, s \rightarrow c : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}.\{N^1_c.K^1_{ac}\}K_{cs}$	D82, D84, modus-ponens
D86	$\Gamma\ 1.5\, c \leftarrow s : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}.\{N^1_c.K^1_{ac}\}K_{cs} \Leftarrow$ $1.4\, s \rightarrow c : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}.\{N^1_c.K^1_{ac}\}K_{cs}$, $4 < 5$	D67, substituição
D87	$\Gamma\ 4 < 5$	D11, substituição
D88	$\Gamma\ 1.4\, s \rightarrow c : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}.\{N^1_c.K^1_{ac}\}K_{cs}$, $4 < 5$	D85, D87, conjunção
D89	$\Gamma\ 1.5\, c \leftarrow s : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}.\{N^1_c.K^1_{ac}\}K_{cs}$	D88, D86, modus-ponens
D90	$\Gamma\ 1.5\, c \lhd \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}.\{N^1_c.K^1_{ac}\}K_{cs} \Leftarrow$ $1.5\, c \leftarrow s : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}.\{N^1_c.K^1_{ac}\}K_{cs}$	D15, substituição
D91	$\Gamma\ 1.5\, c \lhd \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}.\{N^1_c.K^1_{ac}\}K_{cs}$	D89, D90, modus-ponens
D92	$\Gamma\ 1.5\, c \lhd \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}.\{N^1_c.K^1_{ac}\}K_{cs} \Leftrightarrow$ $1.5\, c \lhd \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}, 1.5\, c \lhd K_{as}.\{N^1_c.K^1_{ac}\}K_{cs}$	D18, substituição
D93	$\Gamma\ 1.5\, c \lhd \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}, 1.5\, c \lhd K_{as}.\{N^1_c.K^1_{ac}\}K_{cs} \Leftarrow$ $1.5\, c \lhd \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}.\{N^1_c.K^1_{ac}\}K_{cs}$	D92, bi-implicação
D94	$\Gamma\ 1.5\, c \lhd \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}, 1.5\, c \lhd K_{as}.\{N^1_c.K^1_{ac}\}K_{cs}$	D91, D93, modus-ponens
D95	$\Gamma\ 1.5\, c \lhd \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}$	D94, conjunção
D96	$\Gamma\ 1.5\, c \lhd \{N^1_c.K^1_{ac}\}K_{cs}$	D93, conjunção
D97	Γ MCC	suposição
D98	$\Gamma\ 1.5\, c \lhd N^1_c.K^1_{ac} \Leftarrow$ $1.5\, c \lhd \{N^1_c.K^1_{ac}\}K_{cs}$, $1.5\, c \lhd K_{cs}$	D97, substituição
D99	$\Gamma\ 1.5\, c \lhd K_{cs}$	D51, substituição
D100	$\Gamma\ 1.5\, c \lhd \{N^1_c.K^1_{ac}\}K_{cs}$, $1.5\, c \lhd K_{cs}$	D96, D99, conjunção
D101	$\Gamma\ 1.5\, c \lhd N^1_c.K^1_{ac}$	D100, D98, modus-ponens

D102	$\Gamma\ 1.5\, c \lhd N^1_c.K^1_{ac} \Leftrightarrow$ $1.5\, c \lhd N^1_c, 1.5\, c \lhd K^1_{ac}$	D18, substituição
D103	$\Gamma\ 1.5\, c \lhd N^1_c, 1.5\, c \lhd K^1_{ac} \Leftarrow$ $1.5\, c \lhd N^1_c.K^1_{ac}$	D102, bi-implicação
D104	$\Gamma\ 1.5\, c \lhd N^1_c, 1.5\, c \lhd K^1_{ac}$	D101, D103, modus-ponens
D105	$\Gamma\ 1.5\, c \lhd K^1_{ac}$	D104, conjunção
D106	$\Gamma\ 1.5\, c_b \rightarrow a : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as} \Leftarrow$ $1.5\, c \lhd \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}$	D64, substituição
D107	$\Gamma\ 1.5\, c_b \rightarrow a : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}$	D95, D106, modus-ponens
D108	$\Gamma\ 1.6\, a \leftarrow c_b : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as} \Leftarrow$ $1.5\, c_b \rightarrow a : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as},$ $5 < 6$	D67, substituição
D109	$\Gamma\ 5 < 6$	D11, substituição
D110	$\Gamma\ 1.5\, c_b \rightarrow a : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as},$ $5 < 6$	D107, D109, conjunção
D111	$\Gamma\ 1.6\, a \leftarrow c_b : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}$	D110, D108, modus-ponens
D112	$\Gamma\ 1.6\, a \leftarrow b : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as} \Leftarrow$ $1.6\, a \leftarrow c_b : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}$	D72, substituição
D113	$\Gamma\ 1.6\, a \leftarrow b : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}$	D111, D112, modus-ponens
D114	Γ P5	suposição
D115	$\Gamma\ 1.7\, a \overset{K^1_{ac}}{\leftrightarrow} b \Leftarrow$ $1.6\, a \leftarrow b : \overset{1}{N}{}^1_a.\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as},$ $1.7\, a \lhd K_{as},$ $1.1\, a \boxtimes \overset{2}{N}{}^1_a,$ $1.1\, a \boxtimes \overset{1}{N}{}^1_a,$ $1.1\, a \overset{\overset{2}{N}{}^1_a}{\sim} b,$ $6 < 7,$ $1 < 6$	D114, substituição
D116	$\Gamma\ 6 < 7$	D11, substituição
D117	$\Gamma\ 1 < 6$	D11, substituição
D118	Γ SI1	suposição
D119	$\Gamma\ 1.7\, a \lhd K_{as}$	D118, substituição
D120	$\Gamma\ 1.6\, a \leftarrow b : \overset{1}{N}{}^1_a\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as},$ $1.7\, a \lhd K_{as}$	D113, D119, conjunção

D121	Γ	$1.6\, a \leftarrow b : \overset{1}{N}{}^1_a\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}$,	D120, D7, conjunção
		$1.7\, a \lhd K_{as}$,	
		$1.1\, a \boxtimes \overset{2}{N}{}^1_a$	
D122	Γ	$1.6\, a \leftarrow b : \overset{1}{N}{}^1_a\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}$,	D121, D5, conjunção
		$1.7\, a \lhd K_{as}$,	
		$1.1\, a \boxtimes \overset{2}{N}{}^1_a$,	
		$1.1\, a \boxtimes \overset{1}{N}{}^1_a$	
D123	Γ	$1.6\, a \leftarrow b : \overset{1}{N}{}^1_a\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}$,	D122, D8, conjunção
		$1.7\, a \lhd K_{as}$,	
		$1.1\, a \boxtimes \overset{2}{N}{}^1_a$,	
		$1.1\, a \boxtimes \overset{1}{N}{}^1_a$,	
		$1.1\, a \overset{2}{\underset{N^1_a}{\sim}} b$	
D124	Γ	$1.6\, a \leftarrow b : \overset{1}{N}{}^1_a\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}$,	D123, D116, conjunção
		$1.7\, a \lhd K_{as}$,	
		$1.1\, a \boxtimes \overset{2}{N}{}^1_a$,	
		$1.1\, a \boxtimes \overset{1}{N}{}^1_a$,	
		$1.1\, a \overset{2}{\underset{N^1_a}{\sim}} b$,	
		$6 < 7$	
D125	Γ	$1.6\, a \leftarrow b : \overset{1}{N}{}^1_a\{\overset{2}{N}{}^1_a.K^1_{ac}\}K_{as}$,	D124, D117, conjunção
		$1.7\, a \lhd K_{as}$,	
		$1.1\, a \boxtimes \overset{2}{N}{}^1_a$,	
		$1.1\, a \boxtimes \overset{1}{N}{}^1_a$,	
		$1.1\, a \overset{2}{\underset{N^1_a}{\sim}} b$,	
		$6 < 7$,	
		$1 < 6$	
D126	Γ	$1.7\, a \overset{K^1_{ac}}{\leftrightarrow} b$	D125, D115, modus-ponens
D127	Γ	DCC	suposição
D128	Γ	$K^1_{ac} \neq K^1_{ab}$	D127, substituição
D129	Γ	$1.7\, a \overset{K^1_{ac}}{\leftrightarrow} b$,	D126, D128, conjunção
		$K^1_{ac} \neq K^1_{ab}$	
D130	Γ	$1.7\, a \overset{K^1_{ac}}{\leftrightarrow} b$,	D129, D105, conjunção
		$K^1_{ac} \neq K^1_{ab}$,	

		$1.5\, c \lhd K^1_{ac}$	
D131	Γ	$\exists\, E, I_1, I_2, CH$	D130, existencial
		$E.I_1\, a \overset{CH}{\leftrightarrow} b,$	
		$CH \neq K^E_{ab},$	
		$E.I_2\, c \lhd CH$	

□

A.3 Teoremas sobre o Protocolo de 3 Passos

Teorema 5.7, página 116:

$$\begin{aligned}\Phi_{3p} \cup \Delta_{ep} \vdash \quad & \exists\, E, I_1, I_2, A, B, C, M \\ & E.I_1\, A \rightarrow B : \{M\}K_A^{-1}, \\ & E.I_2\, C \lhd M, \\ & C \neq B\end{aligned}$$

Demonstração.

D1	Γ P1	suposição
D2	Γ $1.1\, a \rightarrow b : \{info\}K_a^{-1}$	D1, substituição
D3	Γ CV	suposição
D4	Γ $1.2\, c \leftarrow a : \{info\}K_a^{-1} \Leftarrow$ $1.1\, a \rightarrow b : \{info\}K_a^{-1}$, $1 < 2$	D3, substituição
D5	Γ MQ	suposição
D6	Γ $1 < 2$	D5, substituição
D7	Γ $1.1\, a \rightarrow b : \{info\}K_a^{-1}, 1 < 2$	D2, D6, conjunção
D8	Γ $1.2\, c \leftarrow a : \{info\}K_a^{-1}$	D7, D4, modus-ponens
D9	Γ PR	suposição
D10	Γ $1.2\, c \lhd \{info\}K_a^{-1} \Leftarrow$ $1.2\, c \leftarrow a : \{info\}K_a^{-1}$	D9, substituição
D11	Γ $1.2\, c \lhd \{info\}K_a^{-1}$	D8, D10, modus-ponens
D12	Γ PER	suposição
D13	Γ $1.2\, c_b \rightarrow a : \{info\}K_a^{-1} \Leftarrow$ $1.2\, c \lhd \{info\}K_a^{-1}$	D12, substituição
D14	Γ $1.2\, c_b \rightarrow a : \{info\}K_a^{-1}$	D11, D13, modus-ponens
D15	Γ TN	suposição
D16	Γ $1.3\, a \leftarrow c_b : \{info\}K_a^{-1} \Leftarrow$ $1.2\, c_b \rightarrow a : \{info\}K_a^{-1}$, $2 < 3$	D15, substituição
D17	Γ $2 < 3$	D5, substituição
D18	Γ $1.2\, c_b \rightarrow a : \{info\}K_a^{-1}$, $2 < 3$	D14, D17, conjunção
D19	Γ $1.3\, a \leftarrow c_b : \{info\}K_a^{-1}$	D18, D16, modus-ponens
D20	Γ RIT	suposição

D21	Γ $1.3\,a \leftarrow b : \{info\}K_a^{-1} \Leftarrow$ $1.3\,a \leftarrow c_b : \{info\}K_a^{-1}$	D20, substituição
D22	Γ $1.3\,a \leftarrow b : \{info\}K_a^{-1}$	D19, D21, modus-ponens
D23	Γ P3	suposição
D24	Γ $1.4\,a \rightarrow b : \langle\{info\}K_a^{-1}\rangle K_a^{-1} \Leftarrow$ $1.3\,a \leftarrow b : \{info\}K_a^{-1}$, $3 < 4$	D23, substituição
D25	Γ $3 < 4$	D5, substituição
D26	Γ $1.3\,a \leftarrow b : \{info\}K_a^{-1}, 3 < 4$	D22, D25, conjunção
D27	Γ $1.4\,a \rightarrow b : \langle\{info\}K_a^{-1}\rangle K_a^{-1}$	D26, D24, modus-ponens
D28	Γ $1.5\,c \leftarrow a : \langle\{info\}K_a^{-1}\rangle K_a^{-1} \Leftarrow$ $1.4\,a \rightarrow b : \langle\{info\}K_a^{-1}\rangle K_a^{-1}$, $4 < 5$	D3, substituição
D29	Γ $4 < 5$	D5, substituição
D30	Γ $1.4\,a \rightarrow b : \langle\{info\}K_a^{-1}\rangle K_a^{-1}, 4 < 5$	D27, D29, conjunção
D31	Γ $1.5\,c \leftarrow a : \langle\{info\}K_a^{-1}\rangle K_a^{-1}$	D30, D28, modus-ponens
D32	Γ $1.5\,c \lhd \langle\{info\}K_a^{-1}\rangle K_a^{-1} \Leftarrow$ $1.5\,c \leftarrow a : \langle\{info\}K_a^{-1}\rangle K_a^{-1}$	D9, substituição
D33	Γ $1.5\,c \lhd \langle\{info\}K_a^{-1}\rangle K_a^{-1}$	D31, D32, modus-ponens
D34	Γ SME	suposição
D35	Γ $1.5\,c \lhd info \Leftarrow$ $1.5\,c \lhd \langle\{info\}K_a^{-1}\rangle K_a^{-1}$, $\langle\{info\}K_a^{-1}\rangle K_a^{-1} \equiv info$	D34, substituição
D36	Γ MD	suposição
D37	Γ $\langle\{info\}K_a^{-1}\rangle K_a^{-1} \equiv info$	D36, substituição
D38	Γ $1.5\,c \lhd \langle\{info\}K_a^{-1}\rangle K_a^{-1}$, $\langle\{info\}K_a^{-1}\rangle K_a^{-1} \equiv info$	D33, D37, conjunção
D39	Γ $1.5\,c \lhd info$	D38, D35, modus-ponens
D40	Γ IU	suposição
D41	Γ $c \neq b$	D40, substituição
D42	Γ $1.1\,a \rightarrow b : \{info\}K_a^{-1}$, $1.5\,c \lhd info$	D2, D39, conjunção
D43	Γ $1.1\,a \rightarrow b : \{info\}K_a^{-1}$, $1.5\,c \lhd info$, $c \neq b$	D42, D41, conjunção
D44	Γ $\exists\, E, I_1, I_2, A, B, C, M$ $E.I_1\, A \rightarrow B : \{M\}K_a^{-1}$, $E.I_2\, C \lhd M$,	D43, existencial

$C \neq B$

□

Teorema 5.8, página 118:

$$\begin{aligned}\Phi'_{3p} \cup \Delta_{ep} \vdash \quad & \exists\, E, I_1, I_2, A, B, C, M \\ & E.I_1\, A \rightarrow B : \{M\}K_A^{-1}, \\ & E.I_2\, C \lhd M, \\ & C \neq B\end{aligned}$$

Demonstração.

D1	Γ P1	suposição
D2	Γ $1.1\, a \rightarrow b : \{info\}K_a^{-1}$	D1, substituição
D3	Γ CV	suposição
D4	Γ $1.2\, c \leftarrow a : \{info\}K_a^{-1} \Leftarrow$ $1.1\, a \rightarrow b : \{info\}K_a^{-1}$, $1 < 2$	D3, substituição
D5	Γ MQ	suposição
D6	Γ $1 < 2$	D5, substituição
D7	Γ $1.1\, a \rightarrow b : \{info\}K_a^{-1}, 1 < 2$	D2, D6, conjunção
D8	Γ $1.2\, c \leftarrow a : \{info\}K_a^{-1}$	D7, D4, modus-ponens
D9	Γ PR	suposição
D10	Γ $1.2\, c \lhd \{info\}K_a^{-1} \Leftarrow$ $1.2\, c \leftarrow a : \{info\}K_a^{-1}$	D9, substituição
D11	Γ $1.2\, c \lhd \{info\}K_a^{-1}$	D8, D10, modus-ponens
D12	Γ PER	suposição
D13	Γ $1^{\|\|}.1\, c_b \rightarrow a : \{info\}K_a^{-1} \Leftarrow$ $1^{\|\|}.1\, c \lhd \{info\}K_a^{-1}$	D12, substituição
D14	Γ PEL	suposição
D15	Γ $1^{\|\|}.1\, c \lhd \{info\}K_a^{-1} \Leftarrow$ $1.2\, c \lhd \{info\}K_a^{-1}$, $1^{\|\|}\|\|1$	D14, substituição
D16	Γ PE	suposição
D17	Γ $1^{\|\|}\|\|1$	D16, substituição
D18	Γ $1.2\, c \lhd \{info\}K_a^{-1}$, $1^{\|\|}\|\|1$	D11, D17, conjunção

D19	$\Gamma\ 1^{\|\|}.1\, c \lhd \{info\}K_a^{-1}$	D18, D15, modus-ponens
D20	$\Gamma\ 1^{\|\|}.1\, c_b \rightarrow a : \{info\}K_a^{-1}$	D19, D13, modus-ponens
D21	Γ TN	suposição
D22	$\Gamma\ 1^{\|\|}.2\, a \leftarrow c_b : \{info\}K_a^{-1} \Leftarrow$ $1^{\|\|}.1\, c_b \rightarrow a : \{info\}K_a^{-1},$ $1 < 2$	D21, substituição
D23	$\Gamma\ 1 < 2$	D5, substituição
D24	$\Gamma\ 1^{\|\|}.1\, c_b \rightarrow a : \{info\}K_a^{-1},$ $1 < 2$	D20, D23, conjunção
D25	$\Gamma\ 1^{\|\|}.2\, a \leftarrow c_b : \{info\}K_a^{-1}$	D24, D22, modus-ponens
D26	Γ RIT	suposição
D27	$\Gamma\ 1^{\|\|}.2\, a \leftarrow b : \{info\}K_a^{-1} \Leftarrow$ $1^{\|\|}.2\, a \leftarrow c_b : \{info\}K_a^{-1}$	D26, substituição
D28	$\Gamma\ 1^{\|\|}.2\, a \leftarrow b : \{info\}K_a^{-1}$	D25, D27, modus-ponens
D29	Γ P2	suposição
D30	$\Gamma\ 1^{\|\|}.3\, a \rightarrow b : \{\{info\}K_a^{-1}\}K_a^{-1} \Leftarrow$ $1^{\|\|}.2\, a \leftarrow b : \{info\}K_a^{-1},$ $2 < 3$	D29, substituição
D31	$\Gamma\ 2 < 3$	D5, substituição
D32	$\Gamma\ 1^{\|\|}.2\, a \leftarrow b : \{info\}K_a^{-1},$ $2 < 3$	D28, D31, conjunção
D33	$\Gamma\ 1^{\|\|}.3\, a \rightarrow b : \{\{info\}K_a^{-1}\}K_a^{-1}$	D32, D30, modus-ponens
D34	$\Gamma\ 1^{\|\|}.4\, c \leftarrow a : \{\{info\}K_a^{-1}\}K_a^{-1} \Leftarrow$ $1^{\|\|}.3\, a \rightarrow b : \{\{info\}K_a^{-1}\}K_a^{-1},$ $3 < 4$	D3, substituição
D35	$\Gamma\ 3 < 4$	D5, substituição
D36	$\Gamma\ 1^{\|\|}.3\, a \rightarrow b : \{\{info\}K_a^{-1}\}K_a^{-1},$ $3 < 4$	D33, D35, conjunção
D37	$\Gamma\ 1^{\|\|}.4\, c \leftarrow a : \{\{info\}K_a^{-1}\}K_a^{-1}$	D36, D34, modus-ponens
D38	$\Gamma\ 1^{\|\|}.4\, c \lhd \{\{info\}K_a^{-1}\}K_a^{-1} \Leftarrow$ $1^{\|\|}.4\, c \leftarrow a : \{\{info\}K_a^{-1}\}K_a^{-1}$	D9, substituição
D39	$\Gamma\ 1^{\|\|}.4\, c \lhd \{\{info\}K_a^{-1}\}K_a^{-1}$	D37, D38, modus-ponens
D40	Γ SI2	suposição
D41	$\Gamma\ \{\{info\}K_a^{-1}\}K_a^{-1} \equiv info$	D40, substituição
D42	Γ SME	suposição
D43	$\Gamma\ 1^{\|\|}.4\, c \lhd info \Leftarrow$ $1^{\|\|}.4\, c \lhd \{\{info\}K_a^{-1}\}K_a^{-1},$ $\{\{info\}K_a^{-1}\}K_a^{-1} \equiv info$	D42, substituição

D44	$\Gamma\ 1^{\|\|}.4\, c \lhd \{\{info\}K_a^{-1}\}K_a^{-1}$, $\{\{info\}K_a^{-1}\}K_a^{-1} \equiv info$	D39, D41, conjunção
D45	$\Gamma\ 1^{\|\|}.4\, c \lhd info$	D44, D43, modus-ponens
D46	$\Gamma\ 1.2\, c \lhd info \Leftarrow$ $1^{\|\|}.4\, c \lhd info$, $1 \| \| 1^{\|\|}$	D14, substituição
D47	$\Gamma\ 1 \| \| 1^{\|\|}$	D16, substituição
D48	$\Gamma\ 1^{\|\|}.4\, c \lhd info$, $1 \| \| 1^{\|\|}$	D45, D47, conjunção
D49	$\Gamma\ 1.2\, c \lhd info$	D48, D46, modus-ponens
D50	Γ IU	suposição
D51	$\Gamma\ c \neq b$	D50, substituição
D52	$\Gamma\ 1.1\, a \rightarrow b : \{info\}K_a^{-1}$, $1.2\, c \lhd info$	D2, D49, conjunção
D53	$\Gamma\ 1.1\, a \rightarrow b : \{info\}K_a^{-1}$, $1.2\, c \lhd info$, $c \neq b$	D52, D51, conjunção
D54	$\Gamma\ \exists\ E, I_1, I_2, A, B, C, M$ $E.I_1\ A \rightarrow B : \{M\}K_A^{-1}$, $E.I_2\ C \lhd M$, $C \neq B$	D53, existencial

□

Continuação do cenário de ataque do protocolo 5.8, página 118:

D55	Γ PLX	suposição
D56	$\Gamma\ 1.2\, c \lhd lixo$	D55, substituição
D57	$\Gamma\ 1.2\, c_b \rightarrow a : lixo \Leftarrow$ $1.2\, c \lhd lixo$	D12, substituição
D58	$\Gamma\ 1.2\, c_b \rightarrow a : lixo$	D56, D57, modus-ponens
D59	$\Gamma\ 1.3\, a \leftarrow c_b : lixo \Leftarrow$ $1.2\, c_b \rightarrow a : lixo$, $2 < 3$	D21, substituição
D60	$\Gamma\ 1.2\, c_b \rightarrow a : lixo$, $2 < 3$	D58, D31, conjunção
D61	$\Gamma\ 1.3\, a \leftarrow c_b : lixo$	D60, D59, modus-ponens
D62	$\Gamma\ 1.3\, a \leftarrow b : lixo \Leftarrow$	D26, substituição

	$1.3\, a \leftarrow c_b : lixo$	
D63	$\Gamma\ 1.3\, a \leftarrow b : lixo$	D61, D62, modus-ponens
D64	Γ P3	suposição
D65	$\Gamma\ 1.4\, a \rightarrow b : \langle lixo \rangle K_a^{-1} \Leftarrow$ $1.3\, a \leftarrow b : lixo,$ $3 < 4$	D64, substituição
D66	$\Gamma\ 1.3\, a \leftarrow b : lixo,$ $3 < 4$	D63, D35, conjunção
D67	$\Gamma\ 1.4\, a \rightarrow b : \langle lixo \rangle K_a^{-1}$	D66, D65, modus-ponens

APÊNDICE B – REGRAS DO CÁLCULO DE SEQÜENTES E ALGORITMO DE APLANAMENTO

Para facilitar a leitura das provas da seção 5, vamos reproduzir nesta seção algumas regras do cálculo de seqüentes utilizadas neste trabalho. Algumas dessas regras advêm de outras mais elementares através de um mecanismo de derivação correto e completo (EBBINGHAUS; FLUM; THOMAS, 1984, cap. 4).

Cada regra (ou um grupo delas) receberá um nome (escrito à sua direita) para futura referência. Usaremos uma sintaxe semelhante a usada por Ebbinghaus, Flum e Thomas (1984, cap. 4), com as seguintes variações sintáticas:

- **,** (vírgula) é o símbolo para a conjunção lógica.
- $\Leftarrow$ é o símbolo para a implicação lógica reversa.
- Ao invés de escrever as premissas de uma regra uma sobre a outra, vamos colocá-las lado a lado, ou seja, a regra:

$$\frac{\begin{array}{l}\Gamma\ \varphi \\ \Gamma\ \psi \Leftarrow \varphi\end{array}}{\Gamma\ \psi}$$

será escrita como:

$$\frac{\Gamma\ \varphi \quad \Gamma\ \psi \Leftarrow \varphi}{\Gamma\ \psi}$$

A expressão $\varphi\frac{t}{x}$ representa a sentença φ tendo todas as ocorrências da variável x substituídas pelo termo t. A letra Γ representa uma seqüência de fórmulas. Lembramos ainda que, por definição, numa linguagem poli-sortida, uma variável ocorrendo numa fórmula só pode ser substituída por um termo de seu mesmo tipo.

$\dfrac{}{\Gamma\ \varphi}$ (suposição), se φ for parte de Γ.	$\dfrac{\Gamma\ \varphi \Leftrightarrow \psi}{\Gamma\ \varphi \Leftarrow \psi}$ ou $\dfrac{\Gamma\ \varphi \Leftrightarrow \psi}{\Gamma\ \psi \Leftarrow \varphi}$ (bi-implicação)
$\dfrac{\Gamma\ \varphi, \psi}{\Gamma\ \varphi}$ ou $\dfrac{\Gamma\ \varphi, \psi}{\Gamma\ \psi}$ ou $\dfrac{\Gamma\ \varphi \quad \Gamma\ \psi}{\Gamma\ \varphi, \psi}$ (conjunção)	$\dfrac{\Gamma\ \phi \quad \Gamma\ \varphi \Leftarrow \phi}{\Gamma\ \varphi}$ (modus-ponens)
$\dfrac{\Gamma\ \varphi}{\Gamma\ \varphi\frac{t}{x}}$ (substituição)	$\dfrac{\Gamma\ \varphi\frac{t}{x}}{\Gamma\ \exists x\varphi}$ (existencial)

Tabela 57: Algumas regras do cálculo de seqüentes usadas neste trabalho

A regra da substituição é aplicada sobre fórmulas que possuem variáveis quantificadas universalmente (EBBINGHAUS; FLUM; THOMAS, 1984, p. 69). Já que esse quantificador está implicitamente presente em todas as fórmulas de nossa linguagem, podemos reescrever a regra usando apenas a notação para fórmula φ, sem o quantificador universal ($\forall$).

A fim de não sobrecarregar as demonstrações de teoremas, consideraremos aplicações sucessivas da regra **existencial** como uma única aplicação, sabendo que o resultado da dedução seria equivalente.

Um conjunto bem maior de regras, bem como o processo de derivação de uma regra a partir de outras regras, são apresentados por Ebbinghaus, Flum e Thomas (1984, cap. 4). Uma outra referência para uma discussão sobre os aspectos de implementação do cálculo de seqüentes é Fitting (1990, cap. 4).

B.1 Algoritmo para Aplanagem Árvores de Provas do Cálculo de Seqüentes

Por motivos de clareza e economia de espaço, preferimos exibir as provas do cálculo de seqüentes de uma maneira diferente da originalmente proposta por Szabo (1969). Vamos definir um algoritmo para *aplanar* árvores de prova, ou seja, torná-las mais simples, semelhantes àquelas provas descritas por Manna (1974).

Definição B.1. *Sejam $\Gamma_i, 1 \leq i \leq n-1$, as seqüências de hipótese e Γ_n a seqüência conclusão de uma árvore de prova. Seja o rótulo inicial de cada seqüência a própria seqüência. Seja $R(\Gamma_i), 1 \leq i \leq n$ o rótulo da seqüência Γi. Para aplanar uma árvore de*

prova da forma:

$$\frac{\begin{array}{cccc}\vdots & \vdots & & \vdots \\ \Gamma_1 & \Gamma_2 & \ldots & \Gamma_{n-1}\end{array}}{\Gamma_n}\ (\textit{regra})$$

Siga o seguinte algoritmo:

começo

1. crie um novo rótulo único para Γ_n, ou seja, $R(\Gamma_n)$ = novo rótulo;
2. **Se** $\Gamma_i, 1 \leq i \leq n-1$ não forem todos folhas **Então**

   aplane as subárvores $\Gamma_i, 1 \leq i \leq n-1$ as quais não forem folhas, em qualquer ordem;

3. escreva uma linha da forma:
   $R(\Gamma_n)$ Γ_n $R(\Gamma_1), R(\Gamma_2), \ldots, R(\Gamma_{n-1})$,regra

fim

Exemplos do uso desse algoritmo são fornecidos no capítulo 5.

APÊNDICE C – CÓDIGO COMPLETO DA ANÁLISE AUTOMATIZADA DO PROTOCOLO DE NESSET

Listamos aqui o código da última tentativa de análise automatizada descrita no capítulo 6, bem como todos os resultados da seguinte consulta ao sistema:

```
?- aprofundamento_iterativo(([E,I] possui [C,k(A,B,1)],C\=A,C\=B),9).
```

As cláusulas comentadas são aquelas que não contribuem para a constatação da falha representada pela sentança acima.

```
% ------------------------------------------
% Analise do Protocolo de Nesset
% Meta-Interpretador com Aprofundamento Iterativo
% e mensagens de tamanho limitado
% ------------------------------------------

% ------------------------------------------
% Meta-Interpretador com Aprofundamento Iterativo
% ------------------------------------------

prova(true,_,_):-
    !.
prova(_,D,Limite):-
    D > Limite,
    !,
```

```
    fail.  %% alcancou a profundidade limite.

prova((A,B),D,Limite):-
    !,
    prova(A,D,Limite),
    prova(B,D,Limite).
prova(A,_,_):-
    predicate_property(A,built_in),
    !,
    call(A).
prova(A,D,Limite):-
    clause(A,B),
    D1 is D+1,
    prova(B,D1,Limite).

aprofundamento_iterativo(G,D):-
    prova(G,0,D).
aprofundamento_iterativo(G,D):-
    write('limite='),
    write(D),
    write(' alcan\c{c}ado. '),
    write('(Tecle <Enter> para continuar.)'),
    get0(C),
    ( C == 10 ->
        D1 is D + 1,
        aprofundamento_iterativo(G,D1) ).

% ---------------------------------------------
% configuracoes do sistema
% ---------------------------------------------

% Declaracao dos operadores infixos

:-op(450,xfy,[#]).
```

```
:-op(500,xfy,[envia,possui,recebe]).

% ------------------------------------------
% Agentes possiveis

agente(a). agente(b). agente(c).

% ------------------------------------------
% Execucoes

execucao(1).
% ------------------------------------------
% Instantes

instante(1). instante(2).

% ------------------------------------------
% Mensagens possiveis

limites_mensagens([1,1,1]). % #, crypt() e ass(), respectivamente

mensagem(M):-
    limites_mensagens([LimSubMsg,LimCryp,LimAss]),
    mensagem(M,[LimSubMsg,LimCryp,LimAss]).

mensagem(X,_):-
    agente(X).

%mensagem(k(X),_):-
%   agente(X).

mensagem(k_1(X),_):-
    agente(X).

mensagem(k(A,B,E),_):-
```

```
    agente(A),
    agente(B),
    execucao(E).

mensagem(nc(A,E),_):-
    agente(A),
    execucao(E).

mensagem(M1#M2,[LimSubMsg,LimCryp,LimAss]):-
    LimSubMsg > 0,
    NovoLimSubMsg is LimSubMsg - 1,
    mensagem(M1,[0,LimCryp,LimAss]),
    mensagem(M2,[NovoLimSubMsg,LimCryp,LimAss]).

mensagem(ass(M,CH),[LimSubMsg,LimCryp,LimAss]):-
    LimAss > 0,
    NovoLimAss is LimAss - 1,
    mensagem(M,[LimSubMsg,LimCryp,NovoLimAss]),
    mensagem(CH,[LimSubMsg,LimCryp,0]).

%mensagem(crypt(M,CH),[LimSubMsg,LimCryp,LimAss]):-
%   LimCryp > 0,
%   NovoLimCryp is LimCryp - 1,
%   mensagem(M,[LimSubMsg,NovoLimCryp,LimAss]),
%   mensagem(CH,[LimSubMsg,0,LimAss]).

% ------------------------------------------
% Conversao de Envio e Recebimento (3 para 4 argumentos)

[E,I] envia [A,A,B,M]:-
    [E,I] envia [A,B,M].

[E,I] envia [A,B,M]:-
    [E,I] envia [A,A,B,M].
```

```
[E,I] recebe [A,B,B,M]:-
    [E,I] recebe [A,B,M].

[E,I] recebe [A,B,M]:-
    [E,I] recebe [A,B,B,M].

%-----------------------------------------
% Axiomas de ambiente
%-----------------------------------------

% Possui chave publica (PCB)

[E,I] possui [A,k(B)]:-
    execucao(E),
    instante(I),
    agente(A),
    agente(B).

% ----------------------------------------
% Possui se recebe (PR)

[E,I] possui [A,M]:-
    execucao(E),
    instante(I),
    agente(A),
    mensagem(M),
    [E,I] recebe [A,_,_,M].

% ----------------------------------------
% Mensagem Assinada (MA)

[E,I] possui [A,M]:-
    execucao(E),
    instante(I),
    agente(A),
```

```
    agente(B),
    mensagem(M),
    [E,I] possui [A,ass(M,k_1(B))],
    [E,I] possui [A,k(B)].

% ---------------------------------------------
% Canal visivel

[E,I2] recebe [A,B,C,M]:-
    execucao(E),
    instante(I1),
    instante(I2),
    agente(A),
    agente(B),
    agente(c),
    mensagem(M),
    I1 < I2,
    [E,I1] envia [B,C,_,M].

% ---------------------------------------------
% Mensagem Simples (MS)

[E,I] possui [A,M2]:-
    execucao(E),
    instante(I),
    agente(A),
    mensagem(M2),
    [E,I] possui [A,_#M2].

%[E,I] possui [A,M1]:-
%    execucao(E),
%    instante(I),
%    agente(A),
%    mensagem(M1),
%    [E,I] possui [A,M1#_].
```

```
%[E,I] possui [A,M1#M2]:-
%    execucao(E),
%    agente(A),
%    mensagem(M1),
%    mensagem(M2),
%    [E,I] possui [A,M1],
%    [E,I] possui [A,M2].

% ---------------------------------------------
% Especificacao do protocolo de Nesset
% ---------------------------------------------

[E,I] envia [A,B,ass(nc(A,E)#k(A,B,E),k_1(A))]:-
    execucao(E),
    instante(I),
    agente(A),
    agente(B).

%[E,I] envia [B,A,crypt(nc(B,E),CH)]:-
%   execucao(E),
%   instante(I),
%   agente(A),
%   agente(B),
%   I2 < 1,
%   [E,I2] recebe [B,A,ass(_#CH,k_1(A))].
```

Abaixo estão os resultados da consulta que demonstra a falha buscada:

```
9 ?- aprofundamento_iterativo(([E,I] possui [C,k(A,B,1)],C\=A,C\=B),9).

E = 1 I = 2 C = a A = b B = b ;

E = 1 I = 2 C = a A = b B = b ;

E = 1 I = 2 C = a A = b B = c ;
```

E = 1 I = 2 C = a A = b B = c ;

E = 1 I = 2 C = a A = c B = b ;

E = 1 I = 2 C = a A = c B = b ;

E = 1 I = 2 C = a A = c B = c ;

E = 1 I = 2 C = a A = c B = c ;

E = 1 I = 2 C = b A = a B = a ;

E = 1 I = 2 C = b A = a B = a ;

E = 1 I = 2 C = b A = a B = c ;

E = 1 I = 2 C = b A = a B = c ;

E = 1 I = 2 C = b A = c B = a ;

E = 1 I = 2 C = b A = c B = a ;

E = 1 I = 2 C = b A = c B = c ;

E = 1 I = 2 C = b A = c B = c ;

E = 1 I = 2 C = c A = a B = a ;

E = 1 I = 2 C = c A = a B = a ;

E = 1 I = 2 C = c A = a B = b ;

E = 1 I = 2 C = c A = a B = b ;

```
E = 1 I = 2 C = c A = b B = a ;

E = 1 I = 2 C = c A = b B = a ;

E = 1 I = 2 C = c A = b B = b ;

E = 1 I = 2 C = c A = b B = b ;
limite=9 alcan\c{c}ado. (Tecle <Enter> para continuar.)
No
10 ?-
```

REFERÊNCIAS

ABADI, M.; NEEDHAM, R. Prudent engineering practice for cryptographic protocols. *IEEE Transactions on Software Engineering*, Los Alamitos, v. 22, n. 1, p. 6–15, Jan. 1996.

ABADI, M.; TUTTLE, M. R. A semantics for a logic of authentication. In: ANNUAL ACM SYMPOSIUM ON PRINCIPALS OF DISTRIBUTED COMPUTING, 10., 1991, Montreal. *Proceedings...* New York: ACM Press, 1991. p. 201–216.

BOYD, C.; MAO, W. Limitations of logical analysis of cryptographic protocols. In: ADVANCES IN CRYPTOLOGY, EUROCRYPT, 1993, Lofthus. *Proceedings...* Berlin: Springer-Verlarg, 1993. p. 240–247.

BOYD, C. A. Hidden assumptions in cryptographic protocols. *IEE Proceedings, Part E*, London, v. 137, n. 6, p. 433–436, Nov. 1990.

BOYD, C. A. A formal framework for authentication. In: EUROPEAN SYMPOSIUM ON RESEARCH IN COMPUTER SECURITY (ESORICS), 2., 1992, Toulouse. *Proceedings...* Berlin: Springer-Verlag, 1992. p. 273–292.

BOYD, C. A. Security architectures using formal methods. *IEEE Jornal on Selected Areas in Communications*, New York, v. 11, n. 5, p. 694–701, June 1993.

BURROWS, M.; ABADI, M.; NEEDHAM, R. *A Logic of Authentication*. Palo Alto, Feb. 1990. Digital Equipment Corporation, Technical Report.

CARLSEN, U. Using logics to detect implementation-dependent flaws. In: ANNUAL COMPUTING SECURITY APPLICATIONS CONFERENCE (ACSAC), 9., 1993, Orlando. *Proceedings...* New York: IEEE Computer Society Press, 1993. p. 64–73.

CARLSEN, U. Cryptographic protocol flaws: Know your enemy. In: IEEE COMPUTER SECURITY FOUNDATIONS WORKSHOP, 7., 1994, Franconia. *Proceedings...* New York: IEEE Computer Society Press, 1994. p. 192–200.

CARLSEN, U. *Formal Specification and Analysis of Cryptographic Protocols*. 168 f. Tese (Doutorado em Eletrônica) — L'Universit Paris XI, Orsay, 1994.

CARLSEN, U. Generating formal cryptographic protocol specifications. In: IEEE SYMPOSIUM ON RESEARCH IN SECURITY AND PRIVACY, 1994, Oakland. *Proceedings...* New York: IEEE Computer Society Press, 1994. p. 137–146.

CASANOVA, M. A.; GIORNO, F. A.; FURTADO, A. L. *Programação em Lógica e a Linguagem PROLOG*. São Paulo: Editora Edgard Blücher Ltda., 1987.

CHAPMAN, D. B.; ZWICKY, E. D. *Building Internet Firewalls*. Sebastopol: O'Reilly & Associates, Inc., 1995.

CHESWICK, W. R.; BELLOVIN, S. M. *Firewalls and Internet Security*: Repelling the Wily Hacker. Reading: Addison-Wesley Publishing Company, 1994.

CLARK, J.; JACOB, J. A Survey of Authentication Protocol Literature. Trabalho em constante desenvolvimento que visa catalogar as publicações da área, incluse com anotações sobre cada publicação. Aug. 1996.

CLOCKSIN, W. F.; MELLISH, C. S. *Programming in Prolog*. 3th rev. and ext. ed. Berlin: Springer-Verlag, 1987.

DECKER, B. D.; PIESSENS, F. Cryptolog: A Theorem Prover for Cryptographic Protocols. In: DIMACS WORKSHOP ON DESIGN AND FORMAL VERIFICATION OF SECURITY PROTOCOLS, 1997, Piscataway. *Proceedings...* Piscataway: Rutgers University, 1997.

DENNING, D. E.; SACCO, G. M. Timestamps in Key Distribution Protocols. *Communications of the ACM*, New York, v. 24, n. 8, p. 533–536, Aug. 1981.

EBBINGHAUS, H.; FLUM, J.; THOMAS, W. *Mathematical Logic*. New York: Springer-Verlag, 1984.

FITTING, M. *First-Order Logic and Automated Theorem Proving*. New York: Springer-Verlag, 1990.

GAREY, M. R.; JOHNSON, D. S. *Computers and Intractability*: A Guide to the Theory of NP-Completeness. Oxford: W.H. Freeman and Company, 1979.

GARFINKEL, S.; SPAFFORD, G. *Practical UNIX and Internet Security*. 2nd exp. and up. ed. Bonn: O'Reilly & Associates, Inc., 1996.

GLIGOR, V. D. et al. Logics for cryptographic protocols: virtues and limitations. In: IEEE COMPUTER SECURITY FOUNDATIONS WORKSHOP, 4., 1991, Franconia. *Proceedings...* Los Alamitos: IEEE Computer Society Press, 1991. p. 219–226.

GONG, L.; NEEDHAM, R. M.; YAHALOM, R. Reasoning About Belief in Cryptographic Protocols. In: *IEEE Computer Society Symposium on Security and Privacy*. Los Alamitos: IEEE Computer Society Press, 1990. p. 234–248.

GRITZALIS, S.; SPINELLIS, D.; GEORGIADIS, P. Security Protocols Over Open Networks and Distributed Systems: Formal Methods for their Analysis, Design, and Verification. *Computer Communications*, New York, v. 22, n. 8, p. 697–709, May 1999.

HEINTZE, N.; TYGAR, J. *Timed Models for Protocol Security*. Pittsburgh, Jan. 1992. School of Computer Science, Carnegie Mellon University, Technical Report.

HOFFMAN, L. J. (Ed.). *Rogue Programs*: Viruses, Worms, and Trojan Horses. New York: Van Nostrand Reinhold, 1990.

HOPCROFT, J. E.; ULLMAN, J. D. *Introduction to Automata Theory, Languages and Computation*. Reading: Addison-Wesley Publishing Company, 1979.

HOROWITZ, E.; SAHNI, S. *Fundamentals of Computer Algorithms*. Potomac: Computer Science Press, Inc., 1978.

HUGHES, J. L. J. *Actually Useful Internet Security Techniques*. Indianapolis: New Riders Publishing, 1995.

HWANG, T.; CHU, H. D.; KU, W. C. A logic-based verification framework for network security protocols. In: INTERNATIONAL JOINT WORKSHOP ON COMPUTER COMMUNICATION (JWCC), 8., 1993, Taipei. *Proceedings...* Taipei, 1993. p. 1–5.

KAUFMAN, C.; PERLMAN, R.; SPECINER, M. *Network Security*: PRIVATE Communication in a PUBLIC World. Englewood Cliffs: Prentice Hall, Inc., 1995.

KELLY, J. J. *The Essence of Logic*. Englewood Cliffs: Prentice Hall, Inc., 1997.

KOCHER, P. C.; FREIER, A. O.; KARLTON, P. *The SSL Protocol (Version 3.0)*. Mountain View, 1996. Netscape Communications, Technical Report.

MANNA, Z. Natural deduction. In: _____. *Mathematical Theory of Computation*. New York: McGraw-Hill, Inc., 1974. p. 108–122.

MAO, W. An augmentation of BAN-like logics. In: IEEE COMPUTER SECURITY FOUNDATIONS WORKSHOP, 8., 1995, Kenmare. *Proceedings...* New York: IEEE Computer Society Press, 1995. p. 44–56.

MAO, W.; BOYD, C. A. Towards formal analysis of security protocols. In: IEEE COMPUTER SECURITY FOUNDATIONS WORKSHOP, 6., 1993, Franconia. *Proceedings...* New York: IEEE Computer Society Press, 1993. p. 147–158.

MAO, W.; BOYD, C. A. Development of authentication protocols: some misconceptions and a new approach. In: IEEE COMPUTER SECURITY FOUNDATIONS WORKSHOP, 7., 1994, Franconia. *Proceedings...* New York: IEEE Computer Society Press, 1994. p. 178–186.

MARRERO, W.; CLARKE, E.; JHA, S. *Model Checking for Security Protocols*. Pittsburgh, May 1997. School of Computer Science, Carnegie Mellon University, Technical Report.

MEADOWS, C. A. A System for the Specification and Analysis of Key Management Protocols. In: IEEE COMPUTER SOCIETY SYMPOSIUM ON SECURITY AND PRIVACY, 1991, Oakland. *Proceedings...* Los Alamitos: IEEE Computer Society Press, 1991. p. 182–195.

MEADOWS, C. A. Applying Formal Methods to the Analysis of a Key Management Protocol. *Journal of Computer Security*, Amsterdam, v. 1, n. 1, p. 5–35, Jan. 1992.

MEADOWS, C. A. Formal verification of cryptographic protocols: A survey. In: ADVANCES IN CRYPTOLOGY – ASIACRYPT: INTERNATIONAL CONFERENCE ON THE THEORY AND APPLICATION OF CRYPTOLOGY, 4., 1994, Wollongong. *Proceedings...* Berlin: Springer-Verlag, 1995. (Lecture Notes in Computer Science), p. 133–150.

MILLEN, J. K.; CLARK, S. C.; FREEDMAN, S. B. The Interrogator: Protocol Security Analysis. *IEEE Transactions on Software Engineering*, Los Alamitos, v. 13, n. 2, p. 247–288, Feb. 1987.

NEEDHAM, R. M.; SCHROEDER, M. D. Using Encryption for Authentication in Large Network of Computers. *Communications of the ACM*, New York, v. 21, n. 12, p. 993–999, Dec. 1978.

NESSET, D. M. A Critique of the Burrow, Abadi and Needham Logic. *ACM Operating Systems Review*, New York, v. 24, n. 2, p. 35–38, Apr. 1990.

O'KEEFE, R. A. *The Craft of Prolog*. Cambridge: The MIT Press, 1990.

OPPLIGER, R. *Authentication Systems for Secure Networks*. Boston: Artech House, Inc., 1996.

RUBIN, A.; HONEYMAN, P. *Formal Methods for the Analysis of Authentication Protocols*. Ann Arbor, Nov. 1993. Center for Information Technology Integration (CITI), University of Michigan, Technical Report.

SCHNEIER, B. *Applied Cryptography*: Protocols, Algorithms, and Source Code in C. 2nd ed. New York: John Wiley & Sons, Inc., 1996.

SHOHAM, Y. *Artificial Intelligence Techniques in Prolog*. Amsterdam: Morgan Kaufmann Publishers, Inc., 1994.

SIMONS, P. Basics of Computer Security: Cryptographic Protocols. Oct. 1996.

STALLINGS, W. *Network and Internetwork Security*: Principles and practice. Englewood Cliffs: Prentice Hall, Inc., 1995.

SYVERSON, P. F.; MEADOWS, C. A. A logical language for specifying cryptographic protocol requirements. In: IEEE COMPUTER SOCIETY SYMPOSIUM ON SECURITY AND PRIVACY, 1993, Oakland. *Proceedings...* New York: IEEE Computer Society Press, 1993. p. 165–177.

SZABO, M. Investigations into logical deduction. In: _____. *The Collected Papers of Gerhard Gentzen*. Amsterdam: North-Holland Publishing Company, 1969. p. 68–128.

TANENBAUM, A. S. Network security. In: _____. *Computer Networks*. 3th ed. Upper Saddle River: Prentice Hall, Inc., 1996. p. 577–622.

WIELEMAKER, J. *SWI-Prolog 3.2*: Reference Manual. Amsterdam, 1999.

WOO, T. Y.; LAM, S. S. Design, verification and implementation of an authentication protocol. In: INTERNATIONAL CONFERENCE ON NETWORK PROTOCOLS, 2., 1994, Boston. *Proceedings...* New York: IEEE Computer Society Press, 1994. p. 81–90.

WOO, T. Y. C.; LAM, S. S. Authentication for distributed systems. *IEEE Computer*, Los Alamitos, v. 25, n. 1, p. 39–52, Jan. 1992.

WOO, T. Y. C.; LAM, S. S. A semantic model for authentication protocols. In: IEEE COMPUTER SOCIETY SYMPOSIUM ON SECURITY AND PRIVACY, 1993, Oakland. *Proceedings...* New York: IEEE Computer Society Press, 1993. p. 178–194.

WOO, T. Y. C.; LAM, S. S. Verifying authentication protocols: Methodology and example. In: INTERNATIONAL CONFERENCE ON NETWORK PROTOCOLS, 1993, San Francisco. *Proceedings...* New York: IEEE Computer Society Press, 1993. p. 36–45.

WOO, T. Y. C.; LAM, S. S. A Lesson on Authentication Protocol Design. *Operating Systems Review*, New York, v. 28, n. 3, p. 24–37, July 1994.