Uma Extensão de Framework de Análise de Protocolos de Composibilidade Universal para Acordo de Chaves com Autenticação Baseado em Identidade

Antonio L. Maia Neto, Ítalo Cunha, Leonardo B. Oliveira

Universidade Federal de Minas Gerais (UFMG)

{lemosmaia,italo,leob}@dcc.ufmg.br

Abstract. *There can be found several frameworks for the security analysis of protocols in the literature based on the modularity of Universal Composability. Although most of them support a variety of cryptosystems, identity-based cryptography is usually not among them. In this paper, we present and prove an extension of a framework to support the identity-based key agreement. We sketch the security analysis of a real protocol for mutual authentication proposed for an IoT context to validate our solution.*

Resumo. *A partir da modularidade da Composibilidade Universal, Küsters et al. propõem um framework onde primitivas criptográficas podem ser utilizadas como módulos fundamentais no projeto de protocolos gerais. Embora o framework contemple diversas primitivas, não há suporte a criptografia baseada em identidade, um cenário emergente de segurança para aplicações de Internet das Coisas. Neste artigo, apresentamos e provamos uma extensão do framework para suportar o acordo de chaves com autenticação baseado em identidade. Nós esboçamos a análise de segurança de um protocolo real de autenticação mútua proposto para o contexto de casas inteligentes para validar nossa extensão.*

1. Introdução

A demonstração de que um protocolo satisfaz as propriedades de segurança para as quais foi projetado é uma etapa fundamental do projeto de protocolos seguros. A construção dessas demonstrações, entretanto, é uma tarefa consideravelmente complexa [Canetti 2001]. As provas de segurança, como são conhecidas, são um longo caminho que se inicia pela especificação de um modelo computacional representativo o suficiente para capturar as ameaças e interferências relevantes do contexto de execução do protocolo, passa pela definição da noção de segurança do modelo, que é a formalização de requisitos que devem ser satisfeitos em uma relação entre os elementos do modelo para garantir a segurança do protocolo, e termina com a redução da noção de segurança às primitivas criptográficas fundamentais do protocolo.

Entre os diferentes métodos de análise segurança de protocolos, os modelos de Composibilidade Universal (*Universal Composability*) [Canetti 2001, Kusters 2006, Hofheinz and Shoup 2015] tem como uma das características principais a modularidade. Neste contexto, protocolos provados seguros nesses modelos podem ser seguramente usados como módulos de outros sistemas maiores em uma operação de composição. Baseando-se nessas características, Küsters e Tuengerthal propõem um

framework de análise de segurança de protocolos onde um conjunto de primitivas criptográficas básicas são previamente provadas e, então, podem ser utilizadas como módulos fundamentais no projeto de protocolos gerais, o que simplifica consideravelmente a análise de segurança de protocolos, eliminando, muitas vezes, a necessidade de complexas reduções [Küsters and Rausch 2017]. Embora o *framework* contemple diversas primitivas básicas, não há suporte, por exemplo, à criptografia baseada em identidade.

A ideia de Criptografia baseada em Identidade (*Identity-based Cryptography – IBC*) foi introduzida no trabalho seminal de Shamir [Shamir 1984] e veio a ser viabilizada, na prática, com o advento de criptografia baseada em emparelhamentos [Sakai et al. 2000]. A principal característica de *IBC* é que a chave pública de uma entidade é derivada de sua identidade que, por sua vez, é univocamente associada a uma informação pública desta entidade dentro do domínio em que está inserida. Portanto, em *IBC* não há necessidade de utilizar certificados digitais e, então, por ser uma alternativa à criptografia de chaves públicas, tem sido utilizada em diferentes cenários onde há dispositivos com restrições computacionais [Oliveira and Dahab 2006, Aranha et al. 2009] ou aplicações de Internet das Coisas (*Internet of Things – IoT*) de forma geral [Salman et al. 2016, Neto et al. 2016, Abbas et al. 2019].

Objetivos. Os objetivos deste trabalho são: estender o *framework* de análise de segurança de protocolos proposto por Küsters e Tuengerthal [Küsters and Rausch 2017] de forma a suportar o acordo de chaves com autenticação baseado em identidade de McCullagh e Barreto [McCullagh and Barreto 2005]; provar nossa extensão com base no problema DBDH (*Decisional Bilinear Diffie-Hellman*); e utilizar os resultados de nosso trabalho para esboçar a análise de segurança de um protocolo real de autenticação mútua e acordo de chaves baseado em identidade proposto para o contexto de *IoT* [Neto et al. 2016].

Organização. O restante do artigo é estruturado da seguinte forma: os trabalhos relacionados são discutidos na Seção 2. Na Seção 3, apresentamos a visão geral do modelo de Composibilidade Universal adotado neste trabalho. Na Seção 4, apresentamos o *framework* de análise de segurança de protocolos que pretendemos estender. Na Seção 5, apresentamos e provamos nossa extensão. Na Seção 6, apresentamos o esboço da análise de um protocolo real do contexto de *IoT*. Concluímos na Seção 7.

2. Trabalhos Relacionados

Como o objetivo principal do nosso trabalho é oferecer ferramental para simplificar a análise de segurança de protocolos em um contexto relevante, descrevemos trabalhos onde estratégias similares foram consideradas.

Canetti e Herzog [Canetti and Herzog 2006] propõem um *framework* de análise simbólica de protocolos de autenticação mútua e de acordo de chaves para criptografia de chaves públicas convencional. Os autores formalizam essas propriedades para funcionalidades ideais no modelo *UC* (*Universally Composable*) [Canetti 2001] de Composibilidade Universal e, então, as traduzem para um protocolo simbólico. A ideia é que protocolos do mundo real propostos para prover autenticação mútua e acordo de chaves possam ser simbolicamente representados se modelados em *UC* e, assim, automaticamente analisados quando comparados com as funcionalidades previamente traduzidas em protocolos simbólicos.

Burmester *et al.* [Burmester et al. 2006] propõem um modelo de Composibilidade Universal construído especificamente para analisar aplicações de identificação por radiofrequência. Com o modelo proposto, os autores analisam a segurança de aplicações reais deste contexto, como os protocolo de autenticação *O-TRAP (Optimistic, Trivial RFID Authentication Protocol)* e *YA-TRAP (Yet Another Trivial Authentication Protocol)*. Em uma extensão do primeiro trabalho, Van Le *et al.* [Van Le et al. 2007] ampliam consideravelmente o modelo, criando um *framework* de suporte à análise de protocolos com as propriedades adicionais de anonimato e *forward-secrecy*, onde os autores propõe e analisam a segurança de dois novos protocolos de autenticação e acordo de chaves.

O *framework* em que baseamos este trabalho foi inicialmente proposto por Küsters e Tuengerthal [Küsters and Tuengerthal 2011], onde os autores proprõem uma funcionalidade ideal de primitivas criptográficas no modelo *IITM* [Kusters 2006] de Composibilidade Universal. Os autores provam a realização da funcionalidade sobre algoritmos baseados em premissas criptográficas padrão com suporte inicial a derivação de chaves simétricas, encriptação simétrica autenticada e não autenticada, geração e verificação de MACs, geração de *nonces* e criptografia de chave públicas, especificamente encriptação e assinatura digital. Em outro trabalho [Küsters and Rausch 2017], Küsters e Rausch estendem o *framework*, aumentando as capacidades de análise de protocolos do mundo real com o suporte ao acordo de chaves de Diffie-Hellman. Para demonstrar a usabilidade do *framework*, os autores fazem a análise de segurança dos protocolos de autenticação mútua e acordo de chaves *ISO 9798-3* e *SIGMA (SIGn-and-MAc)*.

3. Composibilidade Universal

O paradigma da simulação, também conhecido como modelo ideal e real, é uma técnica de análise de segurança de protocolos baseada na ideia de que, para mostrar que um protocolo real $\mathcal{P}$ provê uma propriedade de segurança específica, deve-se, antes, conceber um protocolo idealizado, chamado funcionalidade ideal $\mathcal{F}$, que implementa a mesma propriedade de segurança de $\mathcal{P}$. No cenário ideal, os usuários fornecem as entradas a $\mathcal{F}$ que, por sua vez, realiza a computação local associada à propriedade de segurança que satisfaz e devolve os resultados a cada usuário, apropriadamente. O protocolo $\mathcal{P}$ será pelo menos tão seguro quanto a funcionalidade ideal $\mathcal{F}$, se qualquer ataque causado por um adversário $\mathcal{A}$ de $\mathcal{P}$ puder ser simulado por um adversário $\mathcal{S}$ de $\mathcal{F}$ [Micali et al. 1987].

Os modelos de Composibilidade Universal (*Universal Composability*), por sua vez, ampliam a capacidade da análise de segurança através da operação de composição de diferentes protocolos, possibilitando a análise modular de protocolos complexos. Para isso, estes modelos adicionam uma entidade, chamada ambiente $\mathcal{E}$, que representa tudo que é externo à execução do protocolo, mas que pode influenciar em sua execução, como, por exemplo, os usuários, a execução concorrente de outros protocolos, ruídos, atacantes, usuários de outros protocolos, etc. Assim, no contexto da análise de segurança, em um modelo de Composibilidade Universal o objetivo é mostrar que para qualquer adversário $\mathcal{A}$ do protocolo $\mathcal{P}$ é possível construir um simulador $\mathcal{S}$ tal que o ambiente $\mathcal{E}$ que interage com as configurações real e ideal não consegue distinguir se interage com o protocolo real $\mathcal{P}$ e o adversário $\mathcal{A}$, ou com a funcionalidade ideal $\mathcal{F}$ e o simulador $\mathcal{S}$ [Canetti 2001, Kusters 2006, Hofheinz and Shoup 2015].

Modelo Computacional. O modelo computacional de Composibilidade Universal que adotamos é o das Máquinas de Turing Interativas Inexauríveis (*Inexhaustible Interactive*

Turing Machines – IITMs) [Kusters 2006] que são máquinas de Turing probabilísticas parametrizadas por um polinômio e com fitas de comunicação de entrada e saída.

Análogo ao código fonte de um programa, uma *IITM* é uma máquina M que possui sua própria programação composta pelos nomes das fitas usadas na comunicação com outras máquinas, variáveis usadas na computação interna e o comportamento a cada mensagem recebida nas fitas de entrada. A instância de uma máquina, por sua vez, é a execução do código da máquina. Um sistema $\mathcal{M}$ de *IITM*s é descrito como $\mathcal{M} = M_1 \mid \cdots \mid M_k$, um conjunto de k máquinas que se comunicam através de fitas de entrada e saída. As fitas que não são usadas para comunicação das máquinas dentro do sistema são chamadas de interfaces externas. As interfaces externas podem ser usadas para comunicação com o tipo específico de sistema de *IITM* que representa adversários e simuladores, quando estas recebem o nome de interfaces de rede, ou para comunicação com outros sistemas, quando recebem o nome de interfaces de entrada e saída. Uma instância de uma máquina do sistema é ativada quando uma mensagem é recebida em uma fita de entrada e desativada com a escrita de uma mensagem em uma fita de saída ou com o bloqueio da execução. Se a fita de saída for uma fita de decisão em um sistema de *IITM*s, o sistema termina sua execução. Caso contrário, a fita de saída em questão é fita de entrada de outra máquina que, consequentemente, tem sua instância ativada. Portanto, através dos sistemas de *IITM*s é possível modelar a computação e comunicação das entidades envolvidas na Composibilidade Universal, ou seja, protocolo real $\mathcal{P}$, adversário $\mathcal{A}$, funcionalidade ideal $\mathcal{F}$, simulador $\mathcal{S}$ e ambiente $\mathcal{E}$.

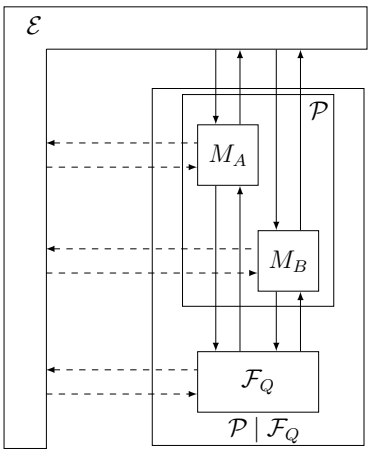
No modelo *IITM*, a noção de segurança, chamada simulabilidade forte (*strong simulatability*), é definida da seguinte forma: sejam o protocolo real $\mathcal{P}$ e a funcionalidade ideal $\mathcal{F}$ sistemas de *IITM*s com interfaces de entrada e saída iguais. Então, $\mathcal{P} \leq \mathcal{F}$, $\mathcal{P}$ realiza $\mathcal{F}$, ou $\mathcal{P}$ é tão seguro quanto $\mathcal{F}$, se existe um simulador $\mathcal{S}$ tal que os sistemas $\mathcal{P}$ e $\mathcal{S} \mid \mathcal{F}$ tem interfaces externas iguais e para qualquer ambiente $\mathcal{E}$ conectado somente à interface externa de $\mathcal{P}$, é verdade que $\mathcal{E} \mid \mathcal{P} \equiv \mathcal{E} \mid \mathcal{S} \mid \mathcal{F}$. Note-se que, em comparação com o objetivo geral de segurança em Composibilidade Universal, na noção de segurança do modelo *IITM* o ambiente $\mathcal{E}$ absorve adversário real $\mathcal{A}$ na representação de equivalência.

O modelo *IITM* provê alguns teoremas de composição para possibilitar a análise modular de sistemas de protocolos. O teorema básico de composição, por exemplo, garante que se n protocolos $\mathcal{P}_1$ ... $\mathcal{P}_n$ realizam, respectivamente, n funcionalidades ideais $\mathcal{F}_1$ ... $\mathcal{F}_n$ ($\mathcal{P}_1 \leq \mathcal{F}_1$ ... $\mathcal{P}_n \leq \mathcal{F}_n$), então a composição dos n protocolos, representada por $\mathcal{P}_1 \mid ... \mid \mathcal{P}_n$, realiza a composição das n funcionalidades, representada por $\mathcal{F}_1 \mid ... \mid \mathcal{F}_n$, ou seja, $\mathcal{P}_1 \mid ... \mid \mathcal{P}_n \leq \mathcal{F}_1 \mid ... \mid \mathcal{F}_n$.

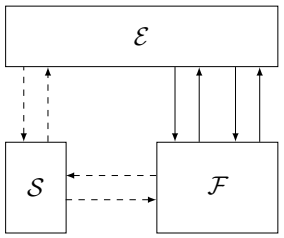
As Figuras 1a e 1b ilustram como o modelo *IITM* é utilizado para modelar computacionalmente a análise de segurança de um protocolo $\mathcal{P} \mid \mathcal{Q}$ de duas partes, A e B, que deve satisfazer as propriedades de segurança representadas por uma funcionalidade ideal $\mathcal{F}$. Na Figura 1a, temos a representação da configuração real. O ambiente é modelado por uma *IITM* única $\mathcal{E}$ e interage com um protocolo composto $\mathcal{P} \mid \mathcal{Q}$. Neste caso, $\mathcal{P}$ é um protocolo real ainda não analisado, modelado por duas máquinas, M_A e M_B, uma para cada parte do protocolo. As máquinas tem interface de entrada e saída, representadas pelas linhas contínuas, para comunicação com os usuários do protocolo, absorvidos em $\mathcal{E}$. O protocolo $\mathcal{Q}$, neste caso, foi previamente provado de forma a realizar a funcionali-

dade ideal $\mathcal{F}_Q$, modelada como uma *IITM* única[1]. As máquinas de $\mathcal{P}$ se comunicam com $\mathcal{F}_Q$ por fitas de entrada e saída. A interface de rede, representada pelas linhas tracejadas, comunicam o adversário, absorvido por $\mathcal{E}$, a M_A, M_B e $\mathcal{F}_Q$.

Na Figura 1b, por sua vez, temos a representação da configuração ideal. O ambiente $\mathcal{E}$ interage com a funcionalidade ideal $\mathcal{F}$ pela interface de entrada e saída, um par de fitas para cada parte em $\mathcal{F}$. O simulador $\mathcal{S}$ interage com $\mathcal{E}$ e $\mathcal{F}$ por interfaces de rede. Neste caso, todas as entidades são modeladas por *IITM*s únicas. Nesta modelagem, para se provar que $\mathcal{P} \mid \mathcal{Q} \leq \mathcal{F}$, deve-se mostrar que $\mathcal{E} \mid \mathcal{P} \mid \mathcal{F}_Q \equiv \mathcal{E} \mid \mathcal{S} \mid \mathcal{F}$.



(a) Modelagem de $\mathcal{E} \mid \mathcal{P} \mid \mathcal{Q}$.



(b) Modelagem de $\mathcal{E} \mid \mathcal{S} \mid \mathcal{F}$.

4. Funcionalidade de primitivas criptográficas $\mathcal{F}_{\text{crypto}}$

Baseando-se nos teoremas de composição do modelo *IITM*, Küsters e Tuengerthal propõem uma funcionalidade ideal de primitivas criptográficas $\mathcal{F}_{\text{crypto}}$ e provam que sua realização $\mathcal{P}_{\text{crypto}}$ pode ser implementada por algoritmos baseados em premissas criptográficas padrão [Küsters and Rausch 2017]. Com esses resultados os autores criam um *framework* que simplifica consideravelmente a análise de protocolos criptográficos reais. Neste contexto, um protocolo real $\mathcal{P}$ pode ser provado realizar uma funcionalidade ideal $\mathcal{F}$ de forma mais simples, a partir da composição $\mathcal{P} \mid \mathcal{F}_{\text{crypto}} \leq \mathcal{F}$, onde as primitivas criptográficas são utilizadas idealmente por $\mathcal{P}$ a partir de $\mathcal{F}_{\text{crypto}}$ e, depois da análise, pelos teoremas do modelo, podem ser diretamente substituídas pela implementação real $\mathcal{P}_{\text{crypto}}$. $\mathcal{F}_{\text{crypto}}$ atualmente suporta as primitivas criptográficas de derivação de chaves, encriptação simétrica autenticada e não autenticada, geração e verificação de MACs, geração de *nonces*, acordo de chaves de Diffie-Hellman (DH) e criptografia de chaves pública, especificamente encriptação e assinatura digital. Como o nosso objetivo é estender a funcionalidade, descrevemos adiante alguns de seus aspectos chave que serão importantes para o entendimento de nossas definições.

$\mathcal{F}_{\text{crypto}}$ é modelada com $2n$ fitas de entrada e saída para comunicação com protocolos de mais alto nível de n partes, um par para cada parte do protocolo. Qualquer usuário de $\mathcal{F}_{\text{crypto}}$ é totalmente identificado pela tupla (pid, $lsid$, r), onde pid é o identificador da parte, $lsid$ é o identificador da sessão local e r é o papel da parte no protocolo.

Para que a segurança possa ser garantida por $\mathcal{F}_{\text{crypto}}$, a funcionalidade não dá aos usuários acesso direto aos elementos secretos. No caso de chaves simétricas e expoentes

[1]Teoremas do modelo permitem provar $\mathcal{P} \mid \mathcal{F}_Q \leq \mathcal{F}$ e, ao final, subtituir $\mathcal{F}_Q$ por $\mathcal{Q}$, concluindo a prova.

de DH, por exemplo, $\mathcal{F}_{\text{crypto}}$ retorna aos usuários apontadores que os referenciam. Assim, a cada operação criptográfica simétrica, o usuário passa o apontador para a chave que deve ser utilizada como parâmetro de entrada na chamada da operação. Por outro lado, as chaves privadas não demandam apontadores, a relação entre o usuário e sua chave é única e controlada internamente pela funcionalidade. Além da representação binária e do apontador, uma chave simétrica em $\mathcal{F}_{\text{crypto}}$ tem uma propriedade *tipo* que modela o propósito único de uso da chave criptográfica. Por exemplo, uma chave do tipo `mac-key` pode ser usada apenas para gerar e verificar MACs.

Como qualquer funcionalidade, $\mathcal{F}_{\text{crypto}}$ também possui a interface de rede para comunicação com o adversário, o simulador. Um aspecto que, a princípio, pode parecer uma contradição é que o simulador provê os elementos secretos de $\mathcal{F}_{\text{crypto}}$. Entretanto, é um atributo de estado de comprometimento do elemento secreto, ou seja, se é conhecido (`known`) ou desconhecido (`unknown`) fora de $\mathcal{F}_{\text{crypto}}$, que determina à funcionalidade se a segurança da operação criptográfica associada ao elemento pode ou não ser garantida. $\mathcal{F}_{\text{crypto}}$ controla o atributo de estado dos elementos secretos através de conjuntos que também são utilizados para modelar a resistência a ataques de repetições, colisões e suposições de elementos desconhecidos.

A funcionalidade $\mathcal{F}_{\text{crypto}}$ é inicializada por uma mensagem na interface externa. Sob tal comando, $\mathcal{F}_{\text{crypto}}$ executa seus algoritmos de inicialização, armazena os parâmetros gerados, envia os resultados ao simulador juntamente com a requisição dos algoritmos criptográficos e pares de chaves dos criptossistemas de chaves públicas. Após receber a resposta do adversário, a inicialização está completa.

5. Extensão e Realização de $\mathcal{F}_{\text{crypto}}$

Nesta seção, descrevemos brevemente o acordo de chaves com autenticação baseado em identidade (Seção 5.1), como modelamos a funcionalidade ideal de primitivas criptográficas $\mathcal{F}_{\text{crypto}}$ para suportar este esquema (Seção 5.2) e como implementamos (Seção 5.3) e provamos (Seção 5.4) a realização de nossa extensão em $\mathcal{P}_{\text{crypto}}$.

5.1. Acordo de Chaves com Autenticação Baseado em Identidade

O acordo de chaves com autenticação baseado em identidade de McCullagh e Barreto, como outros criptossistemas baseados em identidade, necessita de um Centro de Geração de Chaves (*Key Generation Centre – KGC)* que é responsável pela inicialização e distribuição das chaves. O *KCG* inicializa o criptossistema com a entrada de um parâmetro de segurança η em um algoritmo $\mathcal{B}_t$ gerador de parâmetros BDH (*Bilinear Diffie-Hellman*) que, então, gera: dois grupos aditivos $\mathbb{G}_1$ e $\mathbb{G}_2$ de ordem q, onde q é primo, identidade $\mathcal{O}$ e geradores, respectivamente, G_1 e G_2; $\mathbb{G}_T$, um grupo multiplicativo de ordem q e identidade 1; $\hat{e} : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$, uma função de emparelhamento bilinear computável e não-degenerada; e um oráculo aleatório $\mathcal{H} : \{0,1\}^* \rightarrow \mathbb{Z}_q^*$. O *KCG*, então, gera aleatoriamente o segredo $s \in \{0,1\}^*$ e a chave pública mestra do criptossistema $\mathrm{s} \cdot G_1$, onde $\mathrm{s} = \mathcal{H}(s)$. Na prática, $\mathbb{G}_1$ e $\mathbb{G}_2$ são implementados usando um grupo de pontos de uma curva elíptica e o grupo $\mathbb{G}_T$ é implementado usando um subgrupo multiplicativo sobe um corpo de extensão [McCullagh and Barreto 2005].

As chaves pública e privada de uma identidade A são computadas como $P_A^{\mathrm{I}} = \mathrm{a} \cdot G_1 + \mathrm{s} \cdot G_1$ e $S_A^{\mathrm{I}} = \mathrm{a}^{-1} \cdot G_2 + \mathrm{s}^{-1} \cdot G_2$, respectivamente, onde $\mathrm{a} = \mathcal{H}(A)$. Assumindo,

então, que A e B tenham chaves distribuídas pelo mesmo KGC, o acordo de chaves autenticadas entre eles é possível desde que: *i)* A e B selecionem aleatoriamente escalares $\mathrm{x}_A, \mathrm{x}_B, \in \mathbb{Z}_q^*$, *ii)* A e B associem os escalares aleatoriamente selecionados à chave pública do interlocutor, ou seja, A calcula $\mathsf{R}_A = \mathrm{x}_A \cdot P_B^{\mathrm{I}}$ e B calcula $\mathsf{R}_B = \mathrm{x}_B \cdot P_A^{\mathrm{I}}$; *iii)* A e B trocam os resultados dos cálculos em um canal que pode ser público; e *iv)* A e B calculam a mesma chave simétrica através de $A : \hat{e}(\mathsf{R}_B, S_A^{\mathrm{I}})^{\mathbf{x}_A}$ e $B : \hat{e}(\mathsf{R}_A, S_B^{\mathrm{I}})^{\mathbf{x}_B}$. A chaves calculadas no passo *iv* são as mesmas pois, baseando-se nas propriedades dos emparelhamento bilineares, tem-se: $\hat{e}(\mathsf{R}_B, S_A^{\mathrm{I}})^{\mathbf{x}_A} = \hat{e}(\mathrm{x}_B \cdot (\mathrm{a} \cdot G_1 + \mathrm{s} \cdot G_1), \mathrm{a}^{-1} \cdot G_2 + \mathrm{s}^{-1} \cdot G_2)^{\mathbf{x}_A} = \hat{e}((\mathrm{a}+\mathrm{s}) \cdot G_1, (\mathrm{a}+\mathrm{s})^{-1} \cdot G_2)^{\mathbf{x}_A \mathbf{x}_B} = \hat{e}(G_1, G_2)^{\mathbf{x}_A \mathbf{x}_B}$.

5.2. Extensão de $\mathcal{F}_{\text{crypto}}$

Em alto nível, estendemos o acordo de chaves com autenticação baseado em identidade em $\mathcal{F}_{\text{crypto}}$ de forma que um usuário pid possa gerar um escalar secreto único x_{pid} associado à identidade de um usuário pid' em uma relação não secreta R_{pid}. Mais adiante, pid pode combinar seu escalar previamente gerado com uma relação $\mathsf{R}_{pid'}$, eventualmente recebida pela rede e que, analogamente, tenha sido criada pelo interlocutor pid' pela associação de um escalar $\mathrm{x}_{pid'}$ e a identidade de pid, para, então, gerar uma nova chave simétrica derivada das identidades pid e pid'. Para representar as chaves estabelecidas pelo acordo de chaves baseado em identidade, adicionamos à $\mathcal{F}_{\text{crypto}}$ o novo tipo de chaves `id-key` e definimos que chaves `id-key` podem ser geradas apenas pelo novo comando `GenIDKey`, descrito mais adiante, e utilizadas apenas para derivar novas chaves de tipos arbitrários.

Para possibilitar a nova primitiva, adicionamos à inicialização de $\mathcal{F}_{\text{crypto}}$, mais especificamente à etapa onde os algoritmos criptográficos e pares de chaves associados aos criptossistemas já suportados são requisitados ao simulador, a requisição dos parâmetros públicos ($\mathbb{G}_1$, q, G_1) do criptossistema baseado em identidade, assim como todos os pares de chaves (P^{I}, S^{I}) distribuídos neste contexto.

Assim como no caso de chaves simétricas, nossa modelagem de escalares não permite que os usuários os acessem diretamente, mas utilizem apontadores que os referenciem. Por outro lado, as relações R entre escalares e identidades, por serem públicas, são diretamente disponibilizadas aos usuários. Adicionamos dois conjuntos a $\mathcal{F}_{\text{crypto}}$ pra controle interno dos escalares e seus estados de comprometimento. O conjunto geral Scalar, que contém todos os escalares controlados pela funcionalidade e o conjunto $\mathsf{Scalar}_{\mathsf{known}} \subseteq \mathsf{Scalar}$, que mantém os escalares conhecidos fora de $\mathcal{F}_{\text{crypto}}$.

Os escalares também são fornecidos a $\mathcal{F}_{\text{crypto}}$ pelo simulador, o que significa que o estado conhecido de um escalar determina que a chave resultante do acordo de chaves que o envolve também é conhecido fora de $\mathcal{F}_{\text{crypto}}$. Sempre que $\mathcal{F}_{\text{crypto}}$ recebe o escalar x do simulador, verifica se $\mathrm{x} \notin \mathsf{Scalars}$, modelando resistência a colisões e ataques de repetição. Se a verificação falha, $\mathcal{F}_{\text{crypto}}$ requisita um novo valor de escalar ao simulador até que a condição seja satisfeita.

Para implementar nossa extensão, incluímos as seguintes operações a $\mathcal{F}_{\text{crypto}}$[2]:

[2]Para simplificar as descrições, como o acordo de chaves faz sentido apenas no contexto do criptossistema baseado em identidade, em todas as requisições a primeira verificação feita pela funcionalidade é se o usuário que faz a requisição ou aquele envolvido na mesma, possuem chaves distribuídas neste criptossistema e, em caso negativo, uma mensagem de erro é retornada.

- Um usuário (pid, $lsid$, r) requisita a $\mathcal{F}_{\text{crypto}}$ um novo escalar associado à identidade do usuário pid' com a mensagem (`NewScalar`, pid'). $\mathcal{F}_{\text{crypto}}$, então, encaminha a solicitação ao simulador e se comporta como descrito na modelagem de escalares. Posteriormente, $\mathcal{F}_{\text{crypto}}$ faz o registro interno para o controle do escalar (pid, `ptr`, x) e da relação do escalar com a chave pública (x, $P^{\text{I}}_{pid'}$, R) e retorna ao usuário a mensagem (`ScalarPointer`, `ptr`, R), que contém o apontador `ptr` que referencia x e a relação $\mathsf{R} = \mathsf{x} \cdot P^{\text{I}}_{pid'}$.
- Um usuário (pid, $lsid$, r) com acesso a um escalar x pelo apontador `ptr` solicita a $\mathcal{F}_{\text{crypto}}$ uma nova chave derivada do escalar e de uma relação R através da mensagem (`GenIDKey`, `ptr`, R). Primeiramente, $\mathcal{F}_{\text{crypto}}$ verifica se $\mathsf{R} \in \mathbb{G}_1$ para manter a consistência com o criptossistema baseado em identidade. Posteriormente, $\mathcal{F}_{\text{crypto}}$ verifica se R é uma relação existente no conjunto de relações registradas em $\mathcal{F}_{\text{crypto}}$ provenientes do comando `NewScalar`, o que assegura que existe um escalar y associado a uma chave pública de identidade que geraram a relação R. $\mathcal{F}_{\text{crypto}}$, então, verifica se já existe uma chave k derivada de x e y. Caso a verificação seja positiva, um apontador `ptr'` é definido de forma a referenciar k e a mensagem (`IDKey`, `ptr'`) é retornada ao usuário. Se a chave ainda não existe, $\mathcal{F}_{\text{crypto}}$ verifica se relação R recebida envolve a chave pública do usuário pid. Caso esta ou alguma das verificações descritas acima falhe, uma mensagem de erro é retornada. Caso contrário, existe um escalar y que foi gerado por um usuário pid' e associado a P^{I}_{pid}. Os escalares x e y podem, então, ser utilizados na derivação de uma nova chave do novo tipo `id-key`. Como a chave é derivada a partir dos escalares, o comprometimento deles, ou seja, o estado conhecido fora de $\mathcal{F}_{\text{crypto}}$, determina que a funcionalidade não pode garantir a segurança da chave que está sendo gerada. Se os estados dos escalares são desconhecidos, $\mathcal{F}_{\text{crypto}}$ requisita ao simulador uma nova chave desconhecida enviando a mensagem (`ProvideIDKey`, `unknown`, x, y) na interface de rede. Para garantir *freshness*, $\mathcal{F}_{\text{crypto}}$ requisita ao simulador uma nova chave k até que $k \notin \mathsf{Keys}$[3]. Caso contrário, i.e., se o estado de qualquer um escalares é conhecido, $\mathcal{F}_{\text{crypto}}$ requisita uma chave conhecida enviando a mensagem (`ProvideIDKey`, `known`, x, y) ao simulador na interface de rede. Neste caso, como a chave é conhecida, $\mathcal{F}_{\text{crypto}}$ deve prevenir suposição de chaves desconhecidas. Para tal, $\mathcal{F}_{\text{crypto}}$ continua a requisitar ao simulador uma nova chave k até que $k \notin \mathsf{Keys} \backslash \mathsf{Keys}_{\mathsf{known}}$.

  Em ambos os casos, $\mathcal{F}_{\text{crypto}}$ marca a chave k como derivada de x e y, e faz com que o apontador `ptr'` referencie k. Ao final, $\mathcal{F}_{\text{crypto}}$ retorna a mensagem (`IDKey`, `ptr'`) ao usuário.
- Um usuário (pid, $lsid$, r) pode requisitar o valor de um escalar com a mensagem (`RetrieveScalar`, `ptr`). $\mathcal{F}_{\text{crypto}}$, então, adiciona o escalar x_{pid} ao conjunto $\mathsf{Scalar}_{\mathsf{known}}$ e retorna a mensagem (`Scalar`, x_{pid}) ao usuário.

5.3. Realização $\mathcal{P}_{\text{crypto}}$ de $\mathcal{F}_{\text{crypto}}$

Nossa realização do acordo de chaves com autenticação baseado em identidade adiciona à implementação da última versão da máquina $\mathcal{P}_{\text{crypto}}$ [Küsters and Rausch 2017] o criptossistema baseado em identidade de McCullagh e Barreto [McCullagh and Barreto 2005]

[3]Em $\mathcal{F}_{\text{crypto}}$, o estado de comprometimento das chaves e resistência à tentativas de suposição de chaves desconhecidas, colisões e ataques de repetição são modelados pelos conjuntos Keys e $\mathsf{Keys}_{\mathsf{known}}$.

descrito na Seção 5.1. Como em $\mathcal{F}_{\text{crypto}}$, $\mathcal{P}_{\text{crypto}}$ também mantém o registro do estado de cada segredo e necessita do tipo de cada uma das chaves simétricas e, portanto, incluímos esta representação para os escalares e adicionamos o novo tipo de chave simétrica `id-key`. Entretanto, como o único objetivo da informação de estado é a aderência à modelagem de comprometimento de $\mathcal{F}_{\text{crypto}}$, $\mathcal{P}_{\text{crypto}}$ não necessita dos conjuntos utilizados em $\mathcal{F}_{\text{crypto}}$. Isto, porque $\mathcal{P}_{\text{crypto}}$ é fundamentado em primitivas criptográficas reais e, consequentemente, elas devem ser provadas matematicamente, onde a resistência a ataques de repetições, colisões e a suposições de elementos desconhecidos não são garantidos apenas por verificação de elementos já existentes ou não dentro de conjuntos, mas por propriedades matemáticas gerais.

A seguir, detalhamos como cada uma das operações definidas na nossa extensão de $\mathcal{F}_{\text{crypto}}$ é, de fato, implementada em $\mathcal{P}_{\text{crypto}}$[4].

- Um usuário (pid, $lsid$, r) requisita a $\mathcal{P}_{\text{crypto}}$ um novo escalar associado à identidade do usuário pid' com a mensagem (`NewScalar`, pid'). $\mathcal{P}_{\text{crypto}}$, então, seleciona aleatoriamente $\mathrm{x}_{pid} \in \mathbb{Z}_q^*$, cria um apontador ptr para referenciar x_{pid}, calcula $\mathsf{R} = \mathrm{x}_{pid} \cdot P^{\mathrm{I}}_{pid'}$ e retorna ao usuário a mensagem (`ScalarPointer`, ptr, R).
- Um usuário (pid, $lsid$, r) com acesso a um escalar x pelo apontador ptr, solicita a $\mathcal{P}_{\text{crypto}}$ uma nova chave derivada de x e de uma relação R através da mensagem (`GenIDKey`, ptr, R). Primeiramente, $\mathcal{P}_{\text{crypto}}$ verifica se $\mathsf{R} \in \mathbb{G}_1$ para manter a consistência com o criptossistema baseado em identidade. Caso a verificação falhe, uma mensagem de erro é retornada. $\mathcal{P}_{\text{crypto}}$, então, calcula $k = \hat{e}(\mathsf{R}, S^{\mathrm{I}}_{pid})^{\mathrm{x}}$, cria um apontador ptr' que referencia k e retorna (`IDKey`, ptr') ao usuário.
- Um usuário (pid, $lsid$, r) requisita a $\mathcal{P}_{\text{crypto}}$ o valor do escalar x_{pid} referenciado por ptr com a mensagem (`RetrieveScalar`, ptr). $\mathcal{P}_{\text{crypto}}$, retorna a mensagem (`Scalar`, x_{pid}) ao usuário.

5.4. Prova $\mathcal{P}_{\text{crypto}} \leq \mathcal{F}_{\text{crypto}}$

Como na prova original [Küsters and Rausch 2017], para mostrar que $\mathcal{P}_{\text{crypto}} \leq \mathcal{F}_{\text{crypto}}$, as primitivas são consideradas uma a uma em uma série de sistemas híbridos[5]. Nesta seção focamos na prova da primitiva que estendemos em $\mathcal{F}_{\text{crypto}}$, ou seja, o acordo de chaves com autenticação baseado em identidade.

Para mostrar que $\mathcal{E} \mid \mathcal{P}_{\text{crypto}} \equiv \mathcal{E} \mid \mathcal{S} \mid \mathcal{F}_{\text{crypto}}$, focamos nossa argumentação na construção do simulador $\mathcal{S}$, adversário ideal de $\mathcal{F}_{\text{crypto}}$. $\mathcal{S}$ é projetado responder de forma adequada às requisições de $\mathcal{F}_{\text{crypto}}$ baseado na simulação interna de $\mathcal{P}_{\text{crypto}}$. Neste contexto, descrevemos o comportamento de $\mathcal{S}$ em relação às operações que adicionamos a $\mathcal{F}_{\text{crypto}}$:

- $\mathcal{S}$ inicia a simulação de $\mathcal{P}_{\text{crypto}}$, incluindo o criptossistema baseado em identidade.
- Como simulador da funcionalidade, $\mathcal{S}$ inicializa $\mathcal{F}_{\text{crypto}}$ que, por sua vez, requisita a $\mathcal{S}$ os parâmetros ($\mathbb{G}_1$, q, G_1) e pares de chaves (P^{I}, S^{I}) do criptossistema baseado em identidade, que são providos por $\mathcal{S}$ a partir da simulação de $\mathcal{P}_{\text{crypto}}$[6].

[4] Assim como fizemos para simplificar as descrições de $\mathcal{F}_{\text{crypto}}$, $\mathcal{P}_{\text{crypto}}$ sempre faz a verificação inicial de pertencimento dos usuários ao criptossistema baseado em identidade e, em caso negativo, retorna uma mensagem de erro.

[5] A partir da equivalência óbvia $\mathcal{E} \mid \mathcal{P}_{\text{crypto}} \equiv \mathcal{E} \mid \mathcal{S} \mid \mathcal{P}_{\text{crypto}}$, criam-se cópias $\mathcal{P}^i_{\text{crypto}}$ de $\mathcal{P}_{\text{crypto}}$ onde partes da implementação real são substituídas pela versão correspondente ideal de $\mathcal{F}_{\text{crypto}}$. A cada substituição deve-se mostrar que $\mathcal{E} \mid \mathcal{P}_{\text{crypto}} \equiv \mathcal{E} \mid \mathcal{S} \mid \mathcal{P}^i_{\text{crypto}}$ até a equivalência final $\mathcal{E} \mid \mathcal{P}_{\text{crypto}} \equiv \mathcal{E} \mid \mathcal{S} \mid \mathcal{F}_{\text{crypto}}$.

[6] Assume-se que há uma fase prévia de distribuição das chaves.

- Quando $\mathcal{F}_{\text{crypto}}$ requisita um escalar, a partir do criptossistema baseado em identidade da simulação de $\mathcal{P}_{\text{crypto}}$, $\mathcal{S}$ responde com x $\in \mathbb{Z}_q^*$, x aleatório.
- Quando $\mathcal{F}_{\text{crypto}}$ requisita uma chave `id-key` com a mensagem (`ProvideIDKey`, `unknown` ou `known` , x, y), a partir do criptossistema baseado em identidade da simulação de $\mathcal{P}_{\text{crypto}}$, $\mathcal{S}$ retorna $k = \hat{e}(G_1, G_2)^{\text{x·y}}$.
- Sempre que $\mathcal{F}_{\text{crypto}}$ rejeita uma resposta de $\mathcal{S}$ para garantir resistência a ataques de repetição e colisões de escalares e chaves, $\mathcal{S}$ bloqueia a execução.

A prova da nossa extensão $\mathcal{P}_{\text{crypto}}$ deve considerar os casos quando $\mathcal{F}_{\text{crypto}}$, para garantir resistência a ataques de repetição e colisões de escalares e chaves, rejeita as respostas de $\mathcal{S}$, ou seja, quando $\mathcal{S}$ bloqueia a execução. Nesses casos, $\mathcal{E}$ vai claramente distinguir entre as execuções das configurações real e ideal. Portanto, fundamentados nas propriedades do criptossistema baseado em identidade de $\mathcal{P}_{\text{crypto}}$ e dado que $\mathcal{S}$ simula internamente $\mathcal{P}_{\text{crypto}}$, temos que provar matematicamente que o bloqueio da execução acontece com probabilidade desprezível.

Na nossa prova, queremos utilizar a dificuldade do problema de decisão DBDH para demonstrar a consistência entre $\mathcal{P}_{\text{crypto}}$ e $\mathcal{F}_{\text{crypto}}$. O problema DBDH tem o mesmo contexto do algoritmo $\mathcal{B}_t$ gerador de parâmetros da Seção 5.1 e é definido como: sejam $\text{a}, \text{b}, \text{c} \in \mathbb{Z}_q^*$ selecionados aleatoriamente e $Z \in \mathbb{G}_T$ e dado o desafio (G_1, G_2, $\text{a} \cdot G_1$, $\text{a} \cdot G_2$, $\text{b} \cdot G_1$, $\text{c} \cdot G_1$, $\text{c} \cdot G_2$, Z), deve-se distinguir se Z é igual a $\hat{e}(G_1, G_2)^{\text{a·b·c}}$ ou um elemento aleatório de $\mathbb{G}_T$. A vantagem de um algoritmo probabilístico que recebe como entrada o desafio é desprezível. Em termos práticos, a vantagem é definida como a diferença entre a probabilidade do algoritmo retornar 0, que é o caso de quando o algoritmo decide que $Z = \hat{e}(G_1, G_2)^{\text{a·b·c}}$, e a probabilidade do algoritmo retornar 1, que é o caso de quando o algoritmo decide que Z é aleatório.

Iniciamos a prova com a análise da geração de um escalar. Primeiramente, assumimos que a probabilidade de que, sob o comando `NewScalar`, o simulador $\mathcal{S}$ provê um escalar x que é, então, rejeitado por $\mathcal{F}_{\text{crypto}}$, é não desprezível. Podemos dizer, então, que um mesmo escalar x é escolhido em um número não desprezível de execuções de $\mathcal{E} \mid \mathcal{S} \mid \mathcal{F}_{\text{crypto}}$. Como o tempo de execução de todas as máquinas é polinomialmente limitado, o número de escalares x gerados durante a execução também é polinomialmente limitado. Isto nos permite construir um adversário polinomial $\mathcal{A}'$ para atacar o problema DBDH. $\mathcal{A}'$ recebe o desafio do problema (G_1, G_2, $\text{a} \cdot G_1$, $\text{a} \cdot G_2$, $\text{b} \cdot G_1$, $\text{c} \cdot G_1$, $\text{c} \cdot G_2$, Z) gerado a partir do criptossistema baseado em identidade da simulação de $\mathcal{P}_{\text{crypto}}$ e deve retornar 0 se $Z = \hat{e}(G_1, G_2)^{\text{a·b·c}}$, ou retornar 1, se Z é um elemento aleatório de $\mathbb{G}_T$. Então, do mesmo criptossistema, $\mathcal{A}'$ gera um escalar $\text{x} \in \mathbb{Z}_q^*$ e verifica se $\text{x} \cdot G_1 = \text{a} \cdot G_1$, $\text{x} \cdot G_1 = \text{b} \cdot G_1$ ou $\text{x} \cdot G_1 = \text{c} \cdot G_1$. Se as três verificações falham, $\mathcal{A}'$ desiste retornando 1. Caso contrário, $\mathcal{A}'$ pode afirmar o resultado correto já que: se $\text{x} \cdot G_1 = \text{a} \cdot G_1$, $\mathcal{A}'$ verifica se $Z = \hat{e}(\text{b} \cdot G_1, \text{c} \cdot G_2)^{\text{a}} = \hat{e}(G_1, G_2)^{\text{a·b·c}}$ e retorna 0 em caso positivo; se $\text{x} \cdot G_1 = \text{b} \cdot G_1$, $\mathcal{A}'$ verifica se $Z = \hat{e}(\text{a} \cdot G_1, \text{c} \cdot G_2)^{\text{b}} = \hat{e}(G_1, G_2)^{\text{a·b·c}}$ e retorna 0 em caso positivo; se $\text{x} \cdot G_1 = \text{c} \cdot G_1$, $\mathcal{A}'$ verifica se $Z = \hat{e}(\text{b} \cdot G_1, \text{a} \cdot G_2)^{\text{c}} = \hat{e}(G_1, G_2)^{\text{a·b·c}}$ e retorna 0 em caso positivo; caso contrário, $\mathcal{A}'$ retorna 1. Como em um número não desprezível de vezes o adversário $\mathcal{A}'$ não vai desistir por ter encontrado $\text{x} = \text{a}$, b ou c, a vantagem de $\mathcal{A}'$ deixa de ser desprezível, contrariando a dificuldade do problema DBDH. Portanto, por contradição, a probabilidade de rejeição de um escalar por $\mathcal{F}_{\text{crypto}}$ é desprezível.

No segundo passo da prova, devemos analisar os casos onde as chaves `id-key` são rejeitadas por $\mathcal{F}_{\text{crypto}}$. A prova é construída com ataques ao problema de decisão DBDH, como mostrado no passo anterior. Entretanto, por restrições de espaço, apresentamos uma argumentação menos formal para mostrar que, de fato, não há rejeições de chaves por parte de $\mathcal{F}_{\text{crypto}}$. No criptossistema baseado em identidade, a geração de escalares $\mathrm{x} \in \mathbb{Z}_q^*$ é aleatória e, além disso, como demonstrado no primeiro passo da prova, sem repetição, ou seja, uniforme. Assim, o produto de escalares também é aleatório. Como este produto é usado como expoente do emparelhamento dos geradores dos grupos para a geração da chaves, as chaves `id-key` serão sempre uniformemente distribuídas sobre $\mathbb{G}_T$, concluindo nossa argumentação. Assim, concluímos que $\mathcal{P}_{\text{crypto}} \leq \mathcal{F}_{\text{crypto}}$.

6. Estudo de Caso

Nesta seção, apresentamos a modelagem *IITM* de um protocolo real de acordo de chaves com autenticação baseado em identidade proposto para o contexto de *IoT* [Neto et al. 2016] utilizando o resultado deste trabalho, nossa extensão em $\mathcal{F}_{\text{crypto}}$.

O protocolo CHAVEDESESSÃO, mostrado na Figura 2, possibilita a dois dispositivos de *IoT* se autenticarem mutuamente e estabelecerem uma chave de sessão baseada em suas identidades A e B, respectivamente. A análise de segurança do protocolo deve, portanto, mostrar que ele é, de fato, um protocolo de acordo de chaves com autenticação mútua. Para isso, os seguintes passos devem ser seguidos: *i)* especificar $\mathcal{F}$, uma funcionalidade ideal de acordo de chaves com autenticação mútua; *ii)* Modelar e descrever a execução do protocolo como $\mathcal{Q} \mid \mathcal{F}_{\text{crypto}}$, ou seja, onde ele é a composição de uma modelagem $\mathcal{Q}$ com $\mathcal{F}_{\text{crypto}}$, previamente provado no modelo; *iii)* Provar a equivalência $\mathcal{E} \mid \mathcal{Q} \mid \mathcal{F}_{\text{crypto}} \equiv \mathcal{E} \mid \mathcal{S} \mid \mathcal{F}$, definindo $\mathcal{S}$ que internamente simule a execução de $\mathcal{Q} \mid \mathcal{F}_{\text{crypto}}$, definida no passo *ii*, e responda adequadamente a $\mathcal{F}$ garantindo a indistinguibilidade entre as configurações real e ideal para $\mathcal{E}$; e *iv)* Provada a equivalência em *iii*, tem-se que $\mathcal{Q} \mid \mathcal{F}_{\text{crypto}} \leq \mathcal{F}$, portanto, pelos teoremas do modelo, pode-se diretamente substituir $\mathcal{F}_{\text{crypto}}$ por $\mathcal{P}_{\text{crypto}}$ e, então, tem-se a implementação real do protocolo como $\mathcal{Q} \mid \mathcal{P}_{\text{crypto}} \leq \mathcal{F}$.

CHAVEDESESSÃO($dispositivo\ A$, $dispositivo\ B$)

1.	A :	$\mathsf{R}_A \coloneqq \mathrm{x}_A \cdot P_B^{\mathrm{I}}$	6.	$B \to A$:	$\mathrm{n}_B, \mathsf{R}_B, \mathsf{cs\_ack}, \mathrm{MAC}(\mathrm{n}_A)_{MK}$
2.	$A \to B$:	$\mathrm{n}_A, \mathsf{R}_A, \mathsf{cs\_req}$	7.	A :	$k_{A,B} \coloneqq \hat{e}(\mathsf{R}_B, S_A^{\mathrm{I}})^{\mathrm{x}_A}$
3.	B :	$\mathsf{R}_B \coloneqq \mathrm{x}_B \cdot P_A^{\mathrm{I}}$	8.	A :	$MK \coloneqq \mathrm{PRF}(\mathrm{n}_B)_{k_{A,B}}$
4.	B :	$k_{A,B} \coloneqq \hat{e}(\mathsf{R}_A, S_B^{\mathrm{I}})^{\mathrm{x}_B}$	9.	$A \to B$:	$\mathsf{cs\_ack}, \mathrm{MAC}(\mathrm{n}_A \mid \mathrm{n}_B)_{MK}$
5.	B :	$MK \coloneqq \mathrm{PRF}(\mathrm{n}_B)_{k_{A,B}}$			

Figura 2. Acordo de chaves baseado em identidade para *IoT* [Neto et al. 2016].

Por restrições de espaço, não vamos mostrar o passo *iii* da análise de segurança do protocolo, onde, baseado em nossos resultados, seria evidente a contribuição do nosso trabalho com a não necessidade de redução à primitiva criptográfica de acordo de chaves baseado em identidade. Mas, para isso, necessitaríamos, entre outras, de descrever detalhadamente a funcionalidade $\mathcal{F}$. Assim, como os passos *i*, *iii* e *iv* já foram exercitados nas seções anteriores para a prova da nossa extensão, utilizaremos o restante da seção para mostrar como é feita a modelagem e descrição da execução do protocolo usando a nossa extensão em $\mathcal{F}_{\text{crypto}}$.

Os papéis do protocolo, ou seja, os dispositivos A e B, são modelados por duas *IITM*s, M_A e M_B, que se comunicam por fitas internas com $\mathcal{F}_{\text{crypto}}$ para utilizar a funcionalidade como provedora de primitivas criptográficas, ou seja, o protocolo é modelado

como $M_A \mid M_B \mid \mathcal{F}_{\text{crypto}}$. As máquinas M_A e M_B possuem interfaces de entrada e saída para comunicação com os usuários (dispositivos) e interfaces de rede para comunicação com o adversário, que é por onde as mensagens do protocolo são trocadas. É importante relembrar que na noção de segurança do modelo *IITM*os usuários e adversário são absorvidos pelo ambiente $\mathcal{E}$. Para utilizar $\mathcal{F}_{\text{crypto}}$, os dispositivos, usuários de M_A e M_B, devem ser identificados como na funcionalidade, portanto, as máquinas tem que controlar a sessão local $lsid$ de cada dispositivo pid que inicia um acordo de chave.

A prova do protocolo é totalmente dependente da funcionalidade $\mathcal{F}$ e, como discutido, não será abordada. Entretanto, é necessário definir uma interface mínima para $\mathcal{F}$, já que $M_A \mid M_B \mid \mathcal{F}_{\text{crypto}}$ deve ter a mesma interface que $\mathcal{F}$. Neste contexto, consideraremos que a ideia geral de $\mathcal{F}$ é tal que: se duas partes requisitam uma chave secreta baseada em suas identidades para se comunicarem, então $\mathcal{F}$ gera e disponibiliza às partes uma chave de sessão única e conhecida somente por elas. A interface mínima de $\mathcal{F}$ é baseada em uma mensagem de início de acordo de chave, onde o dispositivo pid indica o outro dispositivo pid' com quem deseja estabelecer uma chave secreta, ao que $\mathcal{F}$ responde com uma forma de pid utilizar a chave (no nosso caso, o apontador que referencia a chave) e, ao final, em uma mensagem de encerramento da sessão, onde $\mathcal{F}$ revoga o acesso de pid à chave. A execução do modelo $M_A \mid M_B \mid \mathcal{F}_{\text{crypto}}$ é detalhada em seguida.

- $\mathcal{E} \rightarrow M_A$ e M_B: Início do acordo de chave.
  - pid requisita a M_A um acordo de chaves com pid'.
  - pid' requisita a M_B um acordo de chaves com pid.
  - Ao receber tais requisições, as máquinas definem as sessões locais dos dispositivos, $lsid$ para pid em M_A e $lsid'$ para pid' em M_B e a identificação das partes com quem eles desejam acordar chaves, (pid', $lsid'$) em M_A e (pid, $lsid$) em M_B. A máquina M_A, então, pode iniciar a execução do protocolo em si. A máquina M_B, por sua vez, será ativada ao receber a primeira mensagem na interface de rede (passo 2 no protocolo).
- $M_A \overrightarrow{\leftarrow} \mathcal{F}_{\text{crypto}}$: Preparação para o envio da primeira mensagem.
  - M_A requisita a $\mathcal{F}_{\text{crypto}}$ um escalar associado à identidade pid' com o comando (`NewScalar, pid'`). $\mathcal{F}_{\text{crypto}}$ retorna $\mathsf{R}_{pid} = \mathrm{x}_{pid} \cdot P^{\mathrm{I}}_{pid'}$ e um apontador ptr para o escalar x_{pid}.
  - M_A requisita um novo *nonce* a $\mathcal{F}_{\text{crypto}}$. $\mathcal{F}_{\text{crypto}}$ retorna n_{pid}[7].
- M_A `--→` $\mathcal{E}$ `--→` M_B: M_A envia $m_1 = (\mathrm{n}_{pid}, \mathsf{R}_{pid}, \mathsf{cs\_req})$ na interface de rede com (pid', $lsid'$) como destinatário[8]. Eventualmente, m_1 é entregue na interface de rede de M_B.
- $M_B \overrightarrow{\leftarrow} \mathcal{F}_{\text{crypto}}$: Preparação para o envio da segunda mensagem.
  - M_B requisita a $\mathcal{F}_{\text{crypto}}$ um escalar associado à identidade pid com o comando (`NewScalar, pid`). $\mathcal{F}_{\text{crypto}}$ retorna $\mathsf{R}_{pid'} = \mathrm{x}_{pid'} \cdot P^{\mathrm{I}}_{pid}$ e um apontador ptr' para o escalar $\mathrm{x}_{pid'}$.
  - M_B solicita uma chave `id-key` baseada nas identidades pid e pid' usando o comando (`GenIDKey`, ptr', R_{pid}). $\mathcal{F}_{\text{crypto}}$ retorna ptr_k, apontador que referencia a chave $k = \hat{e}(G_1, G_2)^{\mathrm{x}_{pid}\mathrm{x}_{pid'}}$.

[7]Notação simplificada de elementos de sessão nas máquinas. Escalares, *nonces*, relações, etc., não são ligados apenas ao pid (x_{pid}, n_{pid},R_{pid}), mas também às sessões locais ($\mathrm{x}_{pid,lsid}$, $\mathrm{n}_{pid,lsid}$,$\mathsf{R}_{pid,lsid}$).

[8]No protocolo, cs_req e cs_req_ack são *strings* de requisição e confirmação.

  - M_B requisita a $\mathcal{F}_{\text{crypto}}$ um novo *nonce* e recebe $\text{n}_{pid'}$.
  - M_B requisita a $\mathcal{F}_{\text{crypto}}$ uma nova chave MAC derivada da chave referenciada por ptr_k e do *nonce* $\text{n}_{pid'}$. $\mathcal{F}_{\text{crypto}}$ retorna ptr_{MK}, apontador para a chave MK derivada.
  - M_B requisita a $\mathcal{F}_{\text{crypto}}$ o MAC de $m_2 = (\text{n}_{pid'}, \text{R}_{pid'}, \text{cs\_ack})$ concatenado com n_{pid}, usando a chave referenciada por ptr_{MK}. $\mathcal{F}_{\text{crypto}}$ retorna o MAC σ_2.
- $M_B \dashrightarrow \mathcal{E} \dashrightarrow M_A$: M_B envia $m_2 \mid \sigma_2$ na interface de rede com $(pid, lsid)$ como destinatário. Eventualmente, $m_2 \mid \sigma_2$ é entregue na interface de rede de M_A.
- $M_A \overrightarrow{\leftarrow} \mathcal{F}_{\text{crypto}}$: Verificações para aceite da chave de sessão.
  - M_A solicita uma chave `id-key` baseada nas identidades pid e pid' usando o comando (`GenIDKey`, ptr, $\text{R}_{pid'}$). $\mathcal{F}_{\text{crypto}}$ retorna ptr_k, apontador que referencia a chave $k = \hat{e}(G_1, G_2)^{\text{X}_{pid}\text{X}_{pid'}}$.
  - M_A requisita a $\mathcal{F}_{\text{crypto}}$ uma nova chave MAC referenciada por ptr_k e do *nonce* $\text{n}_{pid'}$. $\mathcal{F}_{\text{crypto}}$ retorna ptr_{MK}, apontador para a chave MK derivada.
  - M_A requisita a $\mathcal{F}_{\text{crypto}}$ a verificação do MAC σ_2 com o MAC de $\text{n}_{pid'}$,$\text{R}_{pid'}$,cs_ack, concatenado com n_{pid}, usando a chave referenciada por ptr_{MK}. Se a verificação for bem-sucedida, dizemos que $(pid, lsid)$ aceitou o acordo de chave. Caso contrário, o protocolo é abortado.
- $M_A \overrightarrow{\leftarrow} \mathcal{F}_{\text{crypto}}$: Preparação para o envio da terceira mensagem.
  - M_A requisita a $\mathcal{F}_{\text{crypto}}$ o MAC de cs_ack concatenado com n_{pid} e $\text{n}_{pid'}$ usando a chave referenciada por ptr_{MK}. $\mathcal{F}_{\text{crypto}}$ retorna o MAC σ_3.
- $M_A \dashrightarrow \mathcal{E} \dashrightarrow M_B$: M_A envia cs_ack $\mid \sigma_3$ na interface de rede com $(pid', lsid')$ como destinatário. Eventualmente, cs_ack $\mid \sigma_3$ é entregue na interface de rede de M_B.
- $M_B \overrightarrow{\leftarrow} \mathcal{F}_{\text{crypto}}$: Verificações para aceite da chave de sessão.
  - M_B requisita a $\mathcal{F}_{\text{crypto}}$ a verificação do MAC σ_3 com o MAC de cs_ack concatenado com n_{pid} e $\text{n}_{pid'}$ usando a chave referenciada por ptr_{MK}. Se a verificação for bem-sucedida, dizemos que $(pid, lsid)$ aceitou o acordo de chave. Caso contrário, o protocolo é abortado.
- M_A e $M_B \rightarrow \mathcal{E}$: Sessão estabelecida. Depois do aceite da chave de sessão, M_A e M_B podem retornar a execução aos dispositivos em $\mathcal{E}$. Neste, caso, como usamos $\mathcal{F}_{\text{crypto}}$, a modelagem demanda que a chave retornada aos usuários seja referenciada pelo apontador ptr_k, o que significa que podemos intermediar operações criptográficas em $\mathcal{F}_{\text{crypto}}$ dentro da sessão, se construirmos $\mathcal{F}$ apropriadamente.
- $\mathcal{E} \rightarrow M_A$ e M_B: Encerramento da sessão. Após utilizar a chave de sessão, os dispositivos pid e pid' enviam mensagens de encerramento de sessão a M_A e M_B, respectivamente. As máquinas, então, removem o acesso dos dispositivos a ptr_k.

7. Conclusão

Neste trabalho, estendemos o *framework IITM* de análise de segurança de protocolos proposto por Küsters e Tuengerthal [Küsters and Rausch 2017] de forma a suportar o acordo de chaves com autenticação baseado em identidade de McCullagh e Barreto [McCullagh and Barreto 2005], provamos nossa extensão com base no problema DBDH e utilizamos nossos resultados para esboçar a análise de segurança do protocolo real de autenticação mútua e acordo de chaves baseado em identidade proposto para o contexto de casas inteligentes [Neto et al. 2016].

Agradecimentos. Os autores agradecem ao CNPq, FAPEMIG e CAPES pelo suporte a este trabalho.

Referências

Abbas, N., Asim, M., Tariq, N., Baker, T., and Abbas, S. (2019). A Mechanism for Securing IoT-enabled Applications at the Fog Layer. *Journal of Sensor and Actuator Networks*.

Aranha, D. F., Oliveira, L. B., Lopez, J., and Dahab, R. (2009). NanoPBC: Implementing Cryptographic Pairings on an 8-bit Platform. In *Conference on Hyperelliptic curves, discrete Logarithms, Encryption, etc (CHiLE 2009)*.

Burmester, M., Van Le, T., and de Medeiros, B. (2006). Provably Secure Ubiquitous Systems: Universally Composable RFID Authentication Protocols. In *(SecureComm'06)*.

Canetti, R. (2001). Universally Composable Security: A New Paradigm for Cryptographic Protocols. In *Symposium on Foundations of Computer Science (FOCS'01)*.

Canetti, R. and Herzog, J. (2006). Universally Composable Symbolic Analysis of Mutual Authentication and Key-Exchange Protocols. In *Theory of Cryptography Conference (TCC'06)*.

Hofheinz, D. and Shoup, V. (2015). GNUC: A New Universal Composability Framework. *Journal of Cryptology*.

Kusters, R. (2006). Simulation-based Security with Inexhaustible Interactive Turing Machines. In *Computer Security Foundations Workshop (CSFW'06)*.

Küsters, R. and Rausch, D. (2017). A Framework for Universally Composable Diffie-Hellman Key Exchange. In *Symposium on Security and Privacy (SP'17)*.

Küsters, R. and Tuengerthal, M. (2011). Ideal Key Derivation and Encryption in Simulation-Based Security. *CT-RSA 2011*.

McCullagh, N. and Barreto, P. S. L. M. (2005). A New Two-party Identity-based Authenticated Key Agreement. In *Conference on Topics in Cryptology (CT-RSA'05)*.

Micali, S., Goldreich, O., and Wigderson, A. (1987). How to Play Any Mental Game. In *Symposium on Theory of Computing (STOC'87)*.

Neto, A. L. M. et al. (2016). AoT: Authentication and Access Control for the Entire IoT Device Life-Cycle. In *Conference on Embedded Network Sensor Systems (Sensys'16)*.

Oliveira, L. B. and Dahab, R. (2006). Pairing-Based Cryptography for Sensor Networks. In *Symposium on Network Computing and Applications (NCA'06)*.

Sakai, R., Ohgishi, K., and Kasahara, M. (2000). Cryptosystems Based on Pairing. In *Symposium on Cryptography and Information Security (SCIS'00)*.

Salman, O., Abdallah, S., Elhajj, I. H., Chehab, A., and Kayssi, A. (2016). Identity-based Authentication Scheme for the Internet of Things. In *Symposium on Computers and Communication (ISCC'16)*.

Shamir, A. (1984). Identity-based Cryptosystems and Signature Schemes. In *Cryptology Conference on Advances in Cryptology (CRYPTO'84)*.

Van Le, T., Burmester, M., and De Medeiros, B. (2007). Universally Composable and Forward-Secure RFID Authentication and Authenticated Key Exchange. In *Symposium on Information, Computer and Communications Security (ASIACCS'07)*.