AdC: um Mecanismo de Controle de Acesso para o Ciclo de Vida das Coisas Inteligentes

Antonio L. Maia Neto[1], Artur Luis Fernandes[1], Italo Cunha[1]
Michele Nogueira[2], Ivan Oliveira Nunes[1], Leonardo Cotta[1]
Nicolas Gentille[3], Antonio A. F. Loureiro[1], Diego F. Aranha[4]
Harsh Kupwade Patil[5], Leonardo B. Oliveira[1]

[1] UFMG [2]UFPR [3]LG Electronics [4]Unicamp [5]LGE Mobile Research
{lemosmaia,arturluis,cunha,ivanolive,leonardo.cotta,loureiro,leob}@dcc.ufmg.br
michele@inf.ufpr.br dfaranha@ic.unicamp.br {nicolas.gentille,harsh.patil}@lge.com

Abstract. *In this paper, we present AdC, a suite of protocols to incorporate authentication and access control for devices in the Internet of Things. AdC relies on cryptographic primitives from the state of the art to enforce an IoT-adequate access control scheme. AdC makes it possible to add new devices to Domestic domains in a wireless and authenticated way. AdC also allows devices from different domains to establish a trust relationship, making possible their inter-operation. To validate our solution, we have developed an AdC prototype for Android smartphones and evaluated the most resource-consuming cryptographic schemes over more constrained devices.*

Resumo. *Nesse artigo, apresentamos Autenticação das Coisas (AdC), um conjunto de protocolos para incorporar autenticação e controle de acesso para dispositivos na Internet das Coisas (IdC). O AdC emprega primitivas criptográficas do estado da arte para implementar um esquema de controle de acesso adequado à IdC. O AdC possibilita que novos dispositivos sejam adicionados à domínios Domésticos de maneira sem fio e autenticada e, além disso, permite que dispositivos de diferentes domínios estabeleçam uma relação de confiança, possibilitando a interoperação deles. Para validar nossa solução, nós desenvolvemos um protótipo do AdC para smartphones Android e avaliamos a execução dos esquemas criptográficos computacionalmente mais caros em dispositivos mais restritos.*

1. Introdução

A Internet das Coisas (IdC) [Ashton 2009] já é parte de nossas vidas. A ideia de um ambiente inteligente formado por elementos computacionais heterogêneos interconectados se tornou realidade. A IdC combina objetos físicos, sensores e atuadores, gerando sistemas ciber-físicos (*e.g.,* cidades, redes e casas inteligentes). Algumas projeções sugerem que, em breve, estaremos cercados por bilhões de dispositivos de IdC.[1]

A autenticação é primordial para a segurança [Stinson 2002]. Os mecanismos de autenticação permitem a distinção entre dados legítimos e forjados, assim como a determinação da origem de uma mensagem [Stinson 2002]. Para IdC, em particular, a autenticação permite controle de acesso, previne ataques *sybil*, e mitiga ataques de negação de serviço. Isto faz da autenticação uma propriedade chave em segurança para IdC [Stinson 2002].

Mesmo que a segurança em IdC tenha recebido atenção da comunidade científica (*e.g.,* [Markmann et al. 2015, Wangham et al. 2013]), as abordagens existentes não

[1] `http://www.gartner.com/newsroom/id/2636073`

atendem às necessidades de autenticação em IdC. Primeiramente, porque os esquemas tradicionais baseados em infraestrutura de chaves públicas (*Public Key Infrastructure* – PKI) e certificados causam sobrecarga significativa de CPU, memória, armazenamento, comunicação e gerenciamento, o que inviabiliza sua utilização em dispositivos de IdC [Stinson 2002]. Em segundo lugar, IdC geralmente requer que dispositivos de um certo domínio sejam capazes de interoperar, de forma segura, com dispositivos que pertençam a diferentes domínios. Acontece que grande parte dos esquemas de autenticação projetados para dispositivos com restrições de recursos (*e.g.,* [Perrig et al. 2001, Zhu et al. 2003, Liu et al. 2005, Oliveira and Dahab 2006, Silva et al. 2013]) assumem que os dispositivos pertencem a um domínio único e, portanto, não podem ser diretamente aplicados a IdC. Por fim, não há, no estado da arte, mecanismo que contemple autenticação durante todo o ciclo de vida de um dispositivo de IdC (*i.e.,* desde a manufatura até o descarte). Assim, há a necessidade de novas soluções, destinadas exclusivamente a atender os requisitos de autenticação para IdC.

Objetivo. Neste artigo, objetivamos projetar, desenvolver e avaliar um esquema de autenticação e controle de acesso para todo o ciclo de vida dos dispositivos de IdC. Nossa solução, Autenticação das Coisas (AdC), é composta por uma família de protocolos criptográficos que provê autenticação e controle de acesso a cada um dos estágios do ciclo de vida de um dispositivo, a saber: *Manufatura*, *Aquisição*, *Implantação*, *Operação* e *Descarte*.

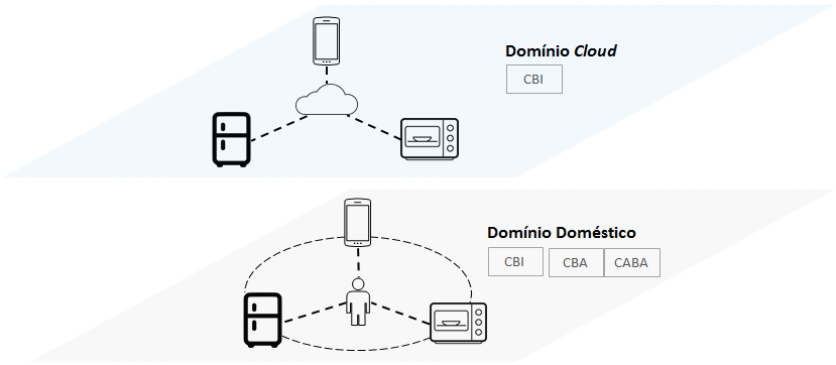


Figura 1. Arquitetura de dois domínios do AdC.

Contribuições. Este trabalho tem as seguintes contribuições:

1. AdC: uma família de protocolos para prover autenticação e controle de acesso durante todo o ciclo de vida de produtos de IdC.
2. Avaliação analítica e experimental do protocolo AdC, assim como a análise, via ferramenta de verificação automática, das propriedades de segurança do AdC. A avaliação experimental quantifica diversas métricas de desempenho em plataformas de hardware com restrição de recursos.
3. Os primeiros resultados experimentais de um esquema de assinatura baseada em atributos já publicados. Nossa implementação, baseada no trabalho de Maji, Prabhakaran e Rosulek [Maji et al. 2008], está disponível publicamente e é eficiente o suficiente para ser executada em dispositivos com restrição de recursos.

Organização. O restante deste artigo é estruturado da seguinte forma: a seção 2 apresenta o protocolo AdC e as premissas sob os quais foi projetado. A seção 3 descreve o desenvolvimento do AdC, com um foco especial à implementação e otimizações de primitivas

criptográficas. A seção 4 avalia a segurança do AdC e os requisitos computacionais necessários para sua implantação. Nós discutimos os trabalhos relacionados na seção 5 e concluímos na seção 6.

2. Autenticação das Coisas

Nesta seção apresentamos Autenticação das Coisas (AdC), uma família de protocolos de autenticação para todas as fases do ciclo de vida de um dispositivo de IdC. A seção 2.1 apresenta a visão geral do AdC, nas seções 2.2–2.7 descrevemos cada um dos protocolos que o compõe e, por fim, discutimos características complementares na seção 2.8.

Premissas. Para o projeto do protocolo assumimos que (i) todas as mensagens são enumeradas e contém o identificador do interlocutor; (ii) material criptográfico pode ser carregado de forma segura nos dispositivos durante o processo de manufatura (iii) PINs podem ser digitados de forma segura nos dispositivos dentro da casa do usuário; e (iv) primitivas criptográficas são ideais – *i.e.,* sem falhas – e podem ser utilizadas como caixas pretas.

2.1. Visão Geral

Problemas. Em IdC, questões elementares como a autenticação e controle de acesso ainda não estão solucionadas. A criptografia de chaves públicas tradicional é computacionalmente mais cara, não adequada para a maior parte dos dispositivos de IdC. Além disso, dada a heterogeneidade dos dispositivos de IdC e seus múltiplos domínios, esquemas de segurança desenvolvidos para outros cenários de redes sem fio e dispositivos com restrições de recursos (e.g., [Perrig et al. 2001, Zhu et al. 2003, Liu et al. 2005, Margi et al. 2009]) não podem ser diretamente aplicados a IdC. Da mesma forma, modelos tradicionais de controle de acesso não são aplicáveis em IdC [Yuan and Tong 2005]. Por exemplo, esquemas de controle de acesso mandatório, discricionário, e baseado em papéis são centrados no usuário e não consideram as relações dinâmicas entre o solicitante e o recurso. Além disso, em redes de larga escala como as de IdC, pode ser inviável gerir as listas de permissão de acesso aos recursos.

Objetivo. Nosso principal objetivo é projetar uma solução de autenticação e controle de acesso adequada para IdC. Neste contexto, consideramos eletrodomésticos inteligentes que interagem entre si, com os dispositivos pessoais dos usuários, com um servidor Doméstico e um servidor *Cloud*, que age como o fabricante dos dispositivos. Nossa solução deve atender às restrições de eficiência e segurança exigidas pelas aplicações de IdC.

Abordagem. Como usual, em nossa abordagem tratamos (i) a distribuição das chaves para a inicialização da segurança e (ii) o controle de acesso para gerenciar as permissões de execução das operações nos dispositivos de IdC. Entretanto, nossa implementação é inovadora: para a distribuição das chaves, aplicamos Criptografia Baseada em Identidade (CBI) [Shamir 1984, Sakai et al. 2000, Boneh and Franklin 2001], enquanto Criptografia Baseada em Atributos (CBA) [Goyal et al. 2006, Bethencourt et al. 2007] é aplicada para implementar criptograficamente o esquema de Controle de Acesso Baseado em Atributos (CABA) [Yuan and Tong 2005], que, por sua vez, gerencia o controle de acesso às operações nos dispositivos.

O principal desafio para adoção de CBI é o conhecido problema da custódia de chaves, onde o Gerador de Chaves Privadas (GCP) pode passar-se por qualquer usuário

no sistema. Para resolver este problema, concebemos uma arquitetura de dois domínios CBI isolados (figura 1): o domínio do fabricante, chamado *Cloud*, e o domínio local, a que chamamos Doméstico. Estes domínios definem as relações de confiança fabricante-dispositivo e (usuário-)(dispositivo-)dispositivo, respectivamente. Não há sobreposição dessas relações e, portanto, material criptográfico gerado no domínio *Cloud* não é válido no domínio Doméstico e vice-versa. Note-se que ainda existe custódia de chaves dentro de cada domínio isoladamente; entretanto, não é considerada mais um problema. A custódia do domínio *Cloud* não influencia as relações domésticas, já que as requisições originadas neste domínio são nulas no domínio Doméstico. Já a custódia de chaves no domínio Doméstico está inserida em um contexto onde o dispositivo que detém as chaves pertence aos mesmos usuários que o gerenciam, evidenciando uma forte relação de confiança.

Considerando-se o controle de acesso das operações nos dispositivos, o CABA simplifica as relações, substituindo permissões discricionárias por políticas baseadas nos atributos dos usuários, o que permite considerar características do recurso e informação de contexto. O CBA, por sua vez, fornece mecanismos criptográficos para adoção do CABA. Mais precisamente, um esquema de assinatura CBA (Assinatura Baseada em Atributos – ABA) pode ser usado para assegurar um subconjunto mínimo de atributos necessários para a execução de uma operação. Assim, as permissões de execução das operações dos dispositivos relacionam, através de políticas de alto nível CABA, os atributos CBA disponíveis no sistema.

Uma das principais características do AdC é que os dispositivos são implantados sem a necessidade de conexões cabeadas. Para tal, o AdC lança mão da proteção física das casas e de um dispositivo confiável (*e.g.,* o *smartphone* do administrador do domínio Doméstico) para servir de ponte durante a primeira comunicação entre o novo dispositivo e o servidor Doméstico e, assim, engendrar a segurança. Esta comunicação pode acontecer sob diferentes tecnologias de codificação e transmissão.

2.2. Protocolos Auxiliares

No AdC, o procedimento auxiliar *ChaveDeSessão* e os protocolos *AcordoDeChaves*, *Distribuição*, *Vinculação* e *Desvinculação* complementam os protocolos dos estágios principais (seções 2.3–2.7) com os seguintes objetivos: derivar chaves de sessão, acordar chaves par a par (*pairwise keys*), distribuir chaves privadas e vincular e desvincular um usuário de um dispositivo no domínio *Cloud*, respectivamente.

O procedimento ChaveDeSessão(k, i) recebe uma chave previamente compartilhada k e um contador i mantido atualizado entre os interlocutores para derivar chaves de sessão. Esta ideia é baseada no trabalho de [Perrig et al. 2001], onde uma função pseudoaleatória é usada para gerar novas chaves.

O protocolo AcordoDeChaves(A, B) é baseado no trabalho de [Sakai et al. 2000]. Ele permite que dois dispositivos (A e B) do mesmo domínio Doméstico troquem chaves par a par empregando emparelhamentos bilineares.

Já no protocolo Distribuição$(D, S, \mathcal{Z}, \mathrm{Y})$, o servidor S do domínio $\mathcal{Z}$ emite a chave privada $S^{\mathrm{Y}}_{D,\mathcal{Z}}$, relativa ao criptossistema Y, para o dispositivo D. A segurança desta operação é garantida pelo uso de uma chave de sessão derivada de uma chave par a par $k_{D,S}$ previamente compartilhada entre o dispositivo D e o servidor S.

Por fim, os protocolos Vinculação(D, U) e Desvinculação(D, U) vincula e des-

vincula, respectivamente, o dispositivo D e o usuário U no domínio *Cloud*.

Em cada um dos protocolos há uma troca de mensagens da forma desafio-resposta entre os interlocutores. Esta estratégia é adotada nos outros protocolos do AdC para assegurar validade (*freshness*) e evitar ataques de repetição.

2.3. *Manufatura*

Manufatura (figura 2) é executado durante a fabricação dos dispositivos. Neste estágio, o material criptográfico do domínio *Cloud* é carregado nos dispositivos. O servidor *Cloud* C gera a identidade e a chave privada do dispositivo D, $id_{D,\mathcal{C}}$ e $S^{\mathrm{I}}_{D,\mathcal{C}}$, respectivamente, no domínio *Cloud* (subscrito $\mathcal{C}$) e criptossistema CBI (sobrescrito I). Posteriormente, C gera uma chave par a par entre C e D ($k_{D,C}$), cria um contador c_D, e os envia ($\rightarrow$), juntamente com o restante do material criptográfico, para o dispositivo D (passo 3). Este envio é feito de forma segura via um canal físico (FIS), garantindo comunicação confidencial e autenticada. Ao final, C apaga $S^{\mathrm{I}}_{D,\mathcal{C}}$ e envia fisicamente ($\Rrightarrow$) D para ser comercializado por seu distribuidor T_D (passo 3).

MANUFATURA(D)
1. C : $id_{D,\mathcal{C}} \coloneqq serial\#\text{de } D,\ S^{\mathrm{I}}_{D,\mathcal{C}}$
2. $C \rightarrow D$: FIS($id_{D,\mathcal{C}}$,$S^{\mathrm{I}}_{D,\mathcal{C}}$,$k_{D,C}$,$\mathrm{c}_D$)
3. $C \Rrightarrow T_D$: D

Figura 2. Estágio *Manufatura*.

AQUISIÇÃO(D, U)
1. $U \rightarrow T_D$: TLS(\$)
2. $T_D \rightarrow C$: TLS(D | U)
3. $C \rightarrow U$: TLS(pin_D)
4. $T_D \Rrightarrow U$: D

Figura 3. Estágio *Aquisição*.

2.4. *Aquisição*

O protocolo *Aquisição* (figura 3) gere a comercialização do dispositivo D, através do distribuidor T_D, para o usuário U. Toda comunicação digital é feita de forma segura por TLS (passos 1, 2 e 3). O usuário U realiza o pedido e paga por D (passo 1). O distribuidor T_D, então, comunica a C a aquisição de D por U (passo 2). C, por sua vez, comunica a U o código de acesso pin_D (passo 3) e, por fim, T_D envia D para U (passo 4).

2.5. *Implantação*

O estágio de *Implantação* (figura 4) inicializa a segurança dos dispositivos em seus domínios Domésticos. Aqui, o usuário *root* (U_r) – provavelmente o dono da casa – e seu dispositivo pessoal D_{U_r} desempenham um papel fundamental. Mais precisamente, D_{U_r} atua como uma ponte de conexão confiável entre o dispositivo que é inserido no domínio Doméstico e o servidor Doméstico H.

Para inserir o novo dispositivo D, U_r digita o código de acesso pin_D em D (passo 1). Em seguida, D obtém de D_{U_r} (i) a identidade $id_{U_r,\mathcal{H}}$ de U_r no domínio Doméstico (subscrito $\mathcal{H}$); (ii) a chave pública $P^{\mathrm{I}}_{H,\mathcal{H}}$ de H, criptossistema CBI do domínio $\mathcal{H}$; (iii) e o contador do domínio $\mathrm{c}_{\mathbb{G}_{\mathrm{H}}}$ (passo 2) que vai prover *freshness* para *broadcasts* ($\Rightarrow$) originados de H. O dispositivo D, subsequentemente, gera, de forma aleatória, uma chave par a par $k_{D,H}$, efêmera entre D e H, a cifra usando a chave pública recém recebida $P^{\mathrm{I}}_{H,\mathcal{H}}$, e envia a cifra resultante ($\mathrm{CIP}(k_{D,H})_{P^{\mathrm{I}}_{H,\mathcal{H}}}$), juntamente com sua identidade $id_{D,\mathcal{H}}$ no domínio $\mathcal{H}$ e as operações suportadas ($info_D$) para D_{U_r} (passo 3). Então, U_r usa D_{U_r} para configurar os atributos $\mathbb{A}_D$ e predicados das operações $\mathbb{Y}_D$ de D (passo 4). Assumimos que a comunicação nesses passos iniciais é feita via canais seguros e adequados para o dispositivo D. Como este protocolo ocorre em um ambiente seguro (*e.g.,* casa do usuário), mecanismos de transmissão como digitar no teclado de um dispositivo ou escanear códigos

de barras poderiam ser cuidadosamente empregados para trocar informações de forma segura.

IMPLANTAÇÃO(D)

1.	$U_r \rightarrow D$:	FIS(pin_D)	8.	H :	$S^{\text{I}}_{D_U,\mathcal{H}}$, $S^{\text{A}}_{D_U,\mathcal{H}}$
2.	$D_{U_r} \rightarrow D$:	FIS($id_{U_r,\mathcal{H}}$,$P^{\text{I}}_{H,\mathcal{H}}$,$\text{c}_{\mathbb{G}_H}$)	9.	D :	DISTRIBUIÇÃO(D, H, $\mathcal{H}$, I)
3.	$D \rightarrow D_{U_r}$:	FIS($id_{D,\mathcal{H}}$,$info_D$, CIP($k_{D,H}$)$_{P^{\text{I}}_{H,\mathcal{H}}}$)	10.	D :	DISTRIBUIÇÃO(D, H, $\mathcal{H}$, A)
4.	$U_r \rightarrow D_{U_r}$:	FIS($\mathbb{A}_D$,$\mathbb{Y}_D$)	11.	$H \Rightarrow \mathbb{G}_H$:	$\mathbb{Y}_{\mathbb{G}_H}$, $\mathbb{A}_{\mathbb{G}_H}$,$\text{c}_{\mathbb{G}_H}$, $\text{SIG}_{S^{\text{I}}_{H,\mathcal{H}}}$
5.	$D_{U_r} \rightarrow H$:	$\text{n}_{D_{U_r}}$, imp_req	12.	D :	VINCULAÇÃO (D, U_r)
6.	$H \rightarrow D_{U_r}$:	n_H, MAC($\text{n}_{D_{U_r}}$)$_{k^i_{D_{U_r},H}}$	13.	$H \rightarrow D_{U_r}$:	imp_ack, MAC($\text{n}_{D_{U_r}}+1$)$_{k^i_{D_{U_r},H}}$
7.	$D_{U_r} \rightarrow H$:	imp, $id_{D,\mathcal{H}}$, $\mathbb{A}_D$, $\mathbb{Y}_D$, $info_D$, CIP($k_{D,H}$)$_{P^{\text{I}}_{H,\mathcal{H}}}$, SIG($\text{n}_H \mid \text{n}_{D_{U_r}}$)$_{S^{\text{I}}_{D_{U_r},\mathcal{H}}}$			

Figura 4. Estágio *Implantação*.

O protocolo prossegue como dispositivo root D_{U_r} requisitando (imp_req) a H, a implantação de D. (Note-se que neste ponto é iniciado um protocolo de desafio-resposta entre D_{U_r} e H, com o uso de números usados somente uma vez, *nonces*, $\text{n}_{D_{U_r}}$ e n_H, Código de Autenticação de Mensagens, MAC, e uma chave de sessão $k^i_{D_{U_r},H}$ entre D_{U_r} e H para assegurar *freshness* e evitar ataques de repetição.) Para esse fim, D_{U_r} encaminha a H o comando de implantação (imp), a informação recebida de D, as configurações inseridas por U_r e a cifra da chave $k_{D,H}$ recentemente gerada. Toda essa informação é enviada de forma autenticada, com a assinatura $\text{SIG}_{S^{\text{I}}_{D_{U_r},\mathcal{H}}}$ de D_{U_r}, utilizando sua chave privada $S^{\text{I}}_{D_{U_r},\mathcal{H}}$ de CBI no domínio $\mathcal{H}$, concluindo o passo 7. Aqui fica evidenciada a ponte de D_{U_r} para autenticação entre D e H. D_{U_r} "assina cegamente" [2] o conteúdo cifrado. Note-se que apesar da dependência de D_{U_r}, D não confia em D_{U_r} para trocar material criptográfico e, por isso, a chave $k_{D,H}$ é cifrada.

Seguindo o protocolo, o servidor Doméstico H deriva (passo 8) as chaves privadas de D no domínio $\mathcal{H}$, relativas a CBI $S^{\text{I}}_{D_U,\mathcal{H}}$ e CBA (sobrescrito A) $S^{\text{A}}_{D_U,\mathcal{H}}$, e as emite (passos 9 e 10). Esta emissão é protegida usando a chave recém compartilhada $k_{D,H}$. Para concluir a inclusão de D, H envia uma mensagem *broadcast* (passo 11), assinada com sua chave privada $S^{\text{I}}_{H,\mathcal{H}}$, para todos os dispositivos domésticos – incluindo o recém inserido D – contendo todo o conjunto de predicados ($\mathbb{Y}_{\mathbb{G}_H}$), atributos ($\mathbb{A}_{\mathbb{G}_H}$) e o contador atualizado do domínio ($\text{c}_{\mathbb{G}_H}$). Nesse ponto, D recebe acesso à Internet, *e.g.,* por meio do roteador Wi-Fi do domínio $\mathcal{H}$. O dispositivo D então se vincula ao usuário U_r no domínio *Cloud* (passo 12). Nós assumimos que D é um dispositivo doméstico, *e.g.,* uma geladeira ou TV, e é, portanto, vinculado a U_r. Se, por outro lado, D for o dispositivo pessoal de um usuário U, D é vinculado a U. O protocolo termina (passo 13) com o servidor H informando a D_{U_r} que D foi inserido com sucesso (imp_ack), concluindo o protocolo de desafio-resposta iniciado no passo 5.

2.6. *Operação*

Operação (figura 5) governa a operação cotidiana entre os dispositivos. Nesse protocolo, o usuário U requisita a execução da operação op no dispositivo B usando o dispositivo A (passo 1). O dispositivo A, então, encaminha a requisição (op_req) ao dispositivo B, que,

[2] Usamos aspas duplas pois assinatura cega se refere a uma construção criptográfica existente [Stinson 2002].

por sua vez, responde com o predicado da operação requisitada Υ_{op}, uma expressão booleana que relaciona atributos do sistema (passo 3). O dispositivo A prova que pode executar a operação com a assinatura da mensagem $\text{SIG}_{S^{\text{A}}_{A,\mathcal{H}}}$ usando sua chave privada CBA $S^{\text{A}}_{A,\mathcal{H}}$ do domínio $\mathcal{H}$. De fato, a chave utilizada para assinatura contém um subconjunto de atributos que deve satisfazer Υ_{op} (passo 4). Se B verifica com sucesso a assinatura, A tem permissões para executar op (passo 5). Note-se que *Operação* também faz uso de um protocolo desafio-resposta (passos 2 a 6), análogo àquele apresentado em *Implantação* (figura 4).

OPERAÇÃO(U, A, B, op)

1. U : usa A para executar op em B
2. $A \rightarrow B$: n_A, op_req
3. $B \rightarrow A$: $\text{n}_B, \Upsilon_{op}, \text{MAC}(\text{n}_A)_{k^i_{A,B}}$
4. $A \rightarrow B$: op, $\text{SIG}(\text{n}_B \mid \text{n}_A)_{S^{\text{A}}_{A,\mathcal{H}}}$
5. B : executa operação op
6. $B \rightarrow A$: op_ack, $\text{MAC}(\text{n}_A + 1)_{k^i_{A,B}}$

Figura 5. Estágio *Operação*.

DESCARTE(U, A, B)

1. A : {*Requisita Descarte*}
2. B : DESVINCULAÇÃO(B, U)
3. B : apaga $S^{\text{I}}_{B,\mathcal{C}}$, $S^{\text{I}}_{B,\mathcal{H}}$, $S^{\text{A}}_{B,\mathcal{H}}$ e $\mathbb{R}_B$
4. B : exibe '*descarte conluído*' na tela

Figura 6. Estágio *Descarte*.

2.7. *Descarte*

De maneira geral, *Descarte* é simplesmente uma operação especial, disponível em todos os dispositivos. Portanto, o protocolo (figura 6) é análogo ao *Operação* (figura 5). Para executar *Descarte*, um usuário U usa o dispositivo A para requisitar a operação de descarte do dispositivo B. A operação é executada se os atributos de A satisfazem o predicado relacionado à essa operação (passos 1–4 do protocolo *Operação* na figura 5). Aqui, assumimos que B pertence à U. Então, B se desvincula do seu dono U (passo 2) e apaga todas suas chaves (passo 3), de ambos os domínios *Cloud* ($S^{\text{I}}_{B,\mathcal{C}}$) e Doméstico ($S^{\text{I}}_{B,\mathcal{H}}$ e $S^{\text{A}}_{B,\mathcal{H}}$), além de todas as chaves simétricas compartilhadas com outros dispositivos (conjunto $\mathbb{R}_B$). O dispositivo B conclui o protocolo exibindo uma mensagem como '*descarte concluído*' em sua tela, que desempenha o papel de uma confirmação (passo 4). Após a execução deste protocolo o dispositivo pode ser descartado.

2.8. Aspectos Complementares

Nesta seção descrevemos algumas características complementares do AdC, em particular, como lidar com transferência de dono de um dispositivo, revogação de chaves e operação inter domínios.

Transferência de dono. Em AdC devemos contemplar a transferência de propriedade de um dispositivo. Esta tarefa é executada pelo protocolo *Transferência*, que é similar ao *Descarte* (figura 6) e as mesmas observações lá feitas são aplicáveis aqui. Contudo, as chaves relacionadas ao domínio *Cloud* não são apagadas. Para mimetizar uma confirmação, o dispositivo deve exibir uma mensagem da forma '*transferência concluída*' em sua tela. A partir daí, o antigo dono do dispositivo pode enviar ao novo dono um código de acesso e, então, entregar o dispositivo ao novo usuário.

Revogação de Chaves. Revogação não é o foco principal do AdC. Apesar disso, como mecanismo de revogação no AdC, nós seguimos o esquema proposto por [Boneh and Franklin 2001]. A estratégia consiste em associar as identidades (chaves públicas) com datas de validade. Assim, as chaves são instantaneamente revogadas assim que a validade expira. A granularidade da validade pode ser definida pelo servidor Doméstico ou *Cloud* (*e.g.,* renovação a cada dia).

Operação Inter Domínios. O AdC define o protocolo *AcordoDeChavesInterDomínio* para possibilitar a interoperação entre um dispositivo de um domínio Doméstico estrangeiro (*i.e.,* um visitante) e um dispositivo do domínio Doméstico local. O protocolo é baseado no trabalho de [McCullagh and Barreto 2005], onde, a partir de um acordo de parâmetros públicos dos respectivos criptossistemas CBI, é possível que dispositivos de diferentes domínios derivem uma mesma chave par a par. O protocolo tem como premissa que a chave pública do domínio Doméstico visitado seja obtida de forma autenticada. Com uma chave par a par estabelecida entre os dispositivos visitante e local, digamos A e B respectivamente, A pode requisitar operações em B. Para tal, A deve executar um protocolo similar ao *Operação* (figura 5). Como A não possui atributos no domínio visitado, o domínio de B deve definir um atributo 'visitante' que possa ser semanticamente atribuído aos dispositivos visitantes, *i.e.,* sem a existência de chaves criptográficas. Desta forma, B permite que A realize alguma operação apenas se o predicado de tal operação é satisfeito pelo atributo 'visitante', após um desafio-resposta baseado em MAC.

3. Desenvolvimento

Nesta seção descrevemos a arquitetura e implementação do protótipo do AdC, apresentando detalhes da camada criptográfica do protocolo.

Arquitetura. A nossa arquitetura é composta por (i) um servidor *Cloud* que estabelece o domínio *Cloud* (fabricante de dispositivos); (ii) múltiplos servidores Domésticos que refletem os diversos domínios Domésticos (*e.g.,* casas dos usuários); e (iii) dispositivos de IdC. O servidor *Cloud* pode fazer alocação de recursos (CPU, armazenamento, memória e banda) sob demanda, cenário típico de ambientes *cloud*. Os servidores Domésticos não necessitam da mesma escalabilidade do servidor *Cloud*, entretanto, assumimos que estão sempre disponíveis e tem capacidade suficiente para controlar centenas de dispositivos. Eles podem ser implantados em consoles de videogames, *desktops*, *gateways* de redes locais ou Módulos de Segurança em Hardware (*Hardware Security Modules* – HSMs). Por fim, os dispositivos se conectam a seus domínios *Cloud* e Doméstico e, em termos de plataforma, podem variar amplamente, inclusive na limitação de recursos.

Implementação. Os servidores *Cloud* e Doméstico são implementados em LAMP (Linux, Apache, MySQL e PHP), enquanto que a implementação dos dispositivos se dá em *smartphones* Android. Os servidores comunicam-se com os dispositivos usando o protocolo HTTP e invocam do PHP os esquemas criptográficos previamente compilados. Nos dispositivos Android, as interfaces de usuário são implementadas utilizando a interface nativa e a chamada das funções criptográficas é feita via Interface Nativa Java (*Java Native Interface* – JNI).

Criptografia. A camada de software criptográfico é construída em linguagem C sobre uma versão estendida da biblioteca criptográfica RELIC [Aranha and Gouvêa]. A implementação é compartilhada por todas as entidades e acessada através da chamada de métodos nativos. A RELIC foi escolhida pelo fato de suportar dispositivos com restrição de recursos (*e.g.,* processadores de 8 e 16 bits e 4KB de RAM) e, também, por implementar de forma eficiente diversos algoritmos criptográficos baseados em curvas elípticas em diferentes níveis de segurança. Nossa versão estendida da biblioteca inclui otimização por meio de código *assembly* elaborado especificamente para a arquitetura ARM.

Em relação ao ABA, esquema chave no mecanismo de controle de acesso do AdC,

nossa implementação é baseada no trabalho de [Maji et al. 2008]. Trata-se da primeira implementação aberta de um esquema de assinaturas para CBA. A implementação é dependente de Criptografia Baseada em Emparelhamentos (CBE) e de *Monotone Spam Programs* – (MSPs), que são matrizes usadas para representar os predicados do AdC como expressões booleanas.

A complexidade computacional dos algoritmos ABA mais executados no AdC, assinatura e verificação (ou sua versão probabilística), depende das operações de CBE, mais precisamente, do produto de emparelhamentos. Essa característica aumenta não apenas a complexidade de tempo mas também o consumo de memória. Para atacar este problema, otimizamos a RELIC para computar produtos de emparelhamentos de forma simultânea. A computação de emparelhamentos pode ser dividida em duas fases: o *laço de Miller*, que consiste de um algoritmo de quadrado e multiplicação e a exponenciação final. Quando um produto de emparelhamentos é computado simultaneamente (operação de multi-emparelhamento), os quadrados na extensão completa do corpo e a exponenciação final podem ser compartilhados para todos os emparelhamentos, mantendo uma variável única para acumular os resultados parciais do laço de Miller. Esta otimização evita o armazenamento de um elemento do grupo multiplicativo $\mathbb{G}_T$ e cerca de 50% de multiplicações de corpo finito por emparelhamento adicional computado no produto.

A nossa implementação do AdC combina, ainda, os seguintes protocolos e algoritmos criptográficos. (i) *Sakai-Ohgishi-Kasahara (SOK)* para acordo de chaves simétricas. (ii) *Boneh-Franklin (CBE)* para cifração CBI (adaptado para utilizar emparelhamentos assimétricos). (iii) *Bellare-Namprempre-Nevem (ABI)* para assinaturas CBI. (iv) *Maji-Prabhakaran-Rosulek (ABA)* como ABA resistente a conluio. (v) *keyed-Hash Message Authentication Code (HMAC)* para MAC (baseado em *hash* SHA256). (vi) *Advanced Encryption Standard (AES)* para cifração simétrica, modo CBC.

Parametrização. Os protocolos criptográficos do AdC foram implementados usando curvas equipadas com emparelhamentos bilineares e com grau de mergulho de 12, suficientes para os níveis de segurança de 80 e 128 bits. Em particular, nós adotamos as curvas [Barreto and Naehrig 2005], parametrizadas pelos inteiros $u = (2^{38} + 2^{32} + 2^5 + 1)$ no nível de 80 bits e $u = -(2^{62} + 2^{55} + 1)$ no nível de 128 bits. As curvas desta família são definidas por um número primo módulo $p = 36u^4 + 36u^3 + 24u^2 + 6u + 1$, ordem $n = 36u^4 + 36u^3 + 18u^2 + 6u + 1$ e suportam uma construção eficiente e otimal de emparelhamentos. Nesta instanciação, os elementos de grupos aditivos tem tamanho 40 e 80 bytes, e do grupo multiplicativo 240 bytes no nível de segurança de 80 bits; e, respectivamente, 64, 128 e 384 bytes no nível de segurança de 128 bits. Multiplicação por escalar de base fixa e variável foram implementadas usando método *comb* de tabela única e exponenciação em janela NAF, respectivamente. Os parâmetros para nível de segurança de 80 bits se destinam a sistemas legados e dispositivos com restrições de recursos, enquanto que parâmetros de segurança de 128 bits são os atuais candidatos a padronização e garantem segurança a longo prazo e eficiência.

4. Avaliação

Nesta seção avaliamos a segurança do AdC utilizando uma ferramenta de verificação automática de protocolos de segurança e mostramos que nossa solução é adequada para dispositivos embarcados com restrição de recursos.

4.1. Segurança

O AdC garante as seguintes propriedades de segurança: (i) autenticação, usando MACs e assinaturas digitais; (ii) confidencialidade, usando cifras; (iii) *freshness*, usando *nonces* e contadores; (iv) integridade, usando MACs, assinaturas digitais e funções de *hash*; e (iv) não-repúdio, usando assinaturas digitais.

Para assegurar a segurança do AdC utilizamos o *Scyther* [Cremers 2008] – uma ferramenta de verificação automática de protocolos de segurança. A ferramenta faz sua análise verificando se um dado protocolo é vulnerável a ataques que poderiam violar certas propriedades de segurança. O *Scyther* assume que as primitivas criptográficas são ideais do ponto de vista de segurança e encontra falhas decorrentes de trocas de mensagens nos protocolos. Todas as propriedades garantidas pelo AdC foram verificadas com sucesso pela ferramenta. O *Scyther* foi escolhido, principalmente, por questões de eficiência e suporte a diferentes classes de ataques [Cremers 2008].

4.2. Avaliação Analítica

Nós avaliamos os custos computacionais, quantificados em função do número de primitivas criptográficas de maior custo – emparelhamentos ($\hat{e}$) e multiplicações de pontos de curvas elípticas por escalar (p)[3] e a sobrecarga de comunicação do AdC, quantidade extra de bytes devido à criptografia. A tabela 1 sumariza os resultados.

Tabela 1. Custo computacional e sobrecarga de comunicação.

	Custo Computacional		Comunicação
Esquema	**Remetente**	**Destinatáio**	**Sobrecarga(bytes)**
ABA [Maji et al. 2008]	$(3+2l+2lt)\hat{e}$	$(2lt+1)p+(lt+t+3)\hat{e}$	$130+65l+129t$
ABA Probabilístico [Maji et al. 2008]	$(3+2l+2lt)\hat{e}$	$(2lt+t+2)p+(l+2)\hat{e}$	$130+65l+129t$
ABI [Cao et al. 2008]	$1p$	$3p$	133
EBI [Boneh and Franklin 2001]	$1\hat{e}+2p$	$1\hat{e}+1p$	129
SOK [Sakai et al. 2000]	$1\hat{e}$	$1\hat{e}$	65

O esquema criptográfico mais custoso é o ABA, cerne do mecanismo de controle de acesso e o mais frequentemente utilizado no AdC. Por isso, as próximas análises serão focadas no ABA. Seu custo computacional cresce com o tamanho das matrizes MSPs, que representam os predicados. Uma MSP tem dimensões $l \times t$, onde l é o número de atributos e t o número de operadores *AND* ($\wedge$) do predicado mais um.

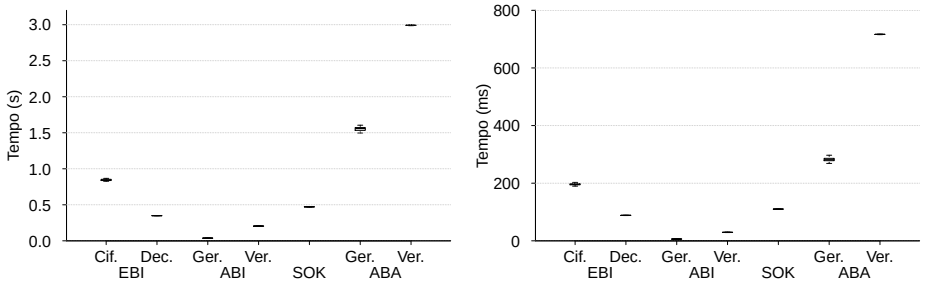
A sobrecarga de comunicação é computada considerando *nonces* de 16 bytes e requisições (req) e confirmações (ack) de 1 byte. No ABA, tal sobrecarga é função do tamanho das MSPs, mas ainda assim, é pequena o suficiente para caber em um ou poucos pacotes de rede.

4.3. Experimentos

Os esquemas criptográficos presentes no AdC foram avaliados em duas plataformas: um Arduino Due (processador ARM M3 de 32 bits e 84 MHz; 96 KB de RAM; 512 KB de memória *flash*) e um Intel Edison (processador Atom de 32 bits e 500 MHz; 1 GB de RAM; 4 GB de memória *flash*). As figuras 7(a) e 7(b) mostram os tempos de execução para: *Cif*ração, *Dec*ifração, *Ger*ação de assinatura, e *Ver*ificação de assinatura nas plataformas Due e Edison, respectivamente. Cada esquema foi executado 100 vezes. Aqui, exibimos os quartis com o 5º e o 95º percentis. Todos esquemas executam em tempo

[3] As outras operações (*e.g.,* primitivas simétricas) são executadas em tempo desprezível.

razoável.[4] Como esperado, o ABA é o esquema mais custoso. No Due são necessários 1,6s para gerar assinaturas e menos de 3,0s para verificá-las, usando predicados da forma $A \wedge B$. Já no Edison estes números chegam a cerca de 300ms e 750ms, respectivamente.



(a) Due (b) Edison

Figura 7. Tempos de execução.

Complementando a figura 7(a), a figura 8 mostra os tempos de execução para o ABA no Due, com predicados da forma $A \wedge B$, diferentes níveis de segurança (80 e 128 bits) e as verificações determinística e probabilística de assinatura. Os resultados mostram um compromisso entre nível de segurança e tempos de execução. Em particular, a verificação probabilística, no nível de 80 bits, permite que assinaturas sejam verificadas em cerca de 1,2s, tempo razoável para a maioria das aplicações interativas. Comparada à sua versão determinística, a abordagem probabilística troca determinismo por eficiência computacional, entretanto, a segurança do algoritmo não é prejudicada por essa troca, a menos de um fator desprezível.

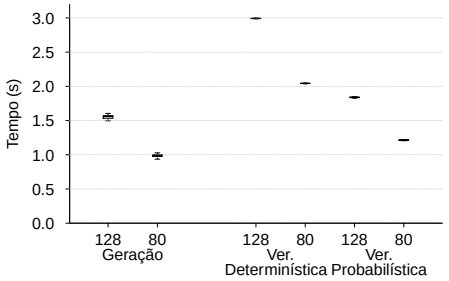


Figura 8. ABA no Due.

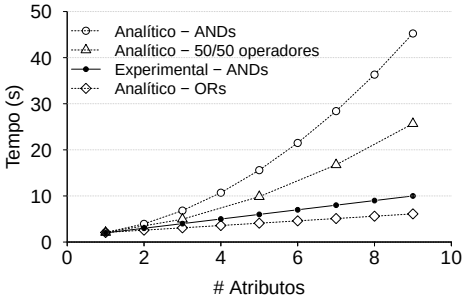


Figura 9. Ver. ABA no Due.

Para verificar como o AdC escala em cenários realísticos de IdC, avaliamos a relação entre tempo de computação e o tamanho dos predicados. A figura 9 mostra os tempos de verificação de assinatura para diferentes predicados. No eixo x representamos a variação no número de atributos. Cada curva corresponde a uma configuração do número de operadores *AND*. Note-se que algumas destas curvas correspondem a tempos de verificação estimados analiticamente. Elas são obtidas da combinação dos tempos de execução das operações de emparelhamento e multiplicação de pontos de curvas elípticas por escalar na plataforma Due (cerca de 355ms e 74,5ms, respectivamente) com os custos analíticos apresentados na tabela 1. A curva experimental da figura 9 mostra que o Due é capaz de verificar assinaturas com predicados complexos com até nove atributos e somente operadores *AND* em cerca de dez segundos. Para predicados mais usuais, entre um e três atributos, a verificação de uma assinatura é feita em menos de seis segundos.

[4]Esquemas simétricos (AES e MAC) foram omitidos pois executaram em menos de 2,5ms.

Pela figura 9 nota-se que os tempos de execução analíticos superestimam significativamente os tempos de execução. Há duas razões principais para isso. Primeiro, nossa implementação é otimizada para computar multi-emparelhamentos simultaneamente (seção 3). Segundo, os custos analíticos consideram que uma matriz MSP de coeficientes $t \times l$ não contém zeros. Na prática, contudo, as matrizes de coeficiente são esparsas, fato que reduz significativamente o número de operações.

Considerando-se o uso de memória do ABA, detectamos a máxima utilização da memória (valor de pico) como o maior tamanho de pilha alcançado durante a execução (a RELIC só aloca memória na pilha). Como esperado, o ABA é adequado mesmo para dispositivos com restrição como o Due, requerendo menos de 19 KB para um predicado com nove atributos e oito operadores *AND*. A utilização de memória deve ser idêntica em outros processadores de 32 bits e um pouco maior em processadores de 64 bits.

Em termos de armazenamento, a implementação da RELIC estendida mais o AdC ocuparam 147 KB da memória do Due, deixando espaço significativo para outras aplicações.

5. Trabalhos Relacionados

Embora a autenticação seja diferente, em termos de conceito e finalidade, de controle de acesso, os dois assuntos são intimamente relacionados. Trabalhos de segurança que discutem dispositivos com limitação de recursos normalmente sugerem o uso de autenticação para estabelecer um mecanismo de controle de acesso. Adiante, descrevemos brevemente trabalhos sobre (i) segurança para dispositivos com limitação de recursos e (ii) esquemas de autenticação e controle de acesso para dispositivos de IdC.

Segurança em dispositivos com limitação de recursos. Antes do advento de IdC, diversos trabalhos discutiram segurança em MANETS e redes de sensores em geral. Os estudos para MANETs (*e.g.,* [Zhou and Haas 1999, Venkatraman and Agrawal 2002]) não são aplicáveis a IdC porque, em sua maioria, consideram a existência de dispositivos com maior poder computacional que aqueles encontrados em IdC. Já os trabalhos destinados a redes de sensores (*e.g.,* [Perrig et al. 2001, Zhu et al. 2003, Liu et al. 2005, Oliveira et al. 2008, Patil and Szygenda 2012]) usualmente se baseiam em premissas que não se aplicam a IdC, logo, essas propostas não podem ser utilizadas. Por exemplo, em uma rede de sensores todos os dispositivos executam a mesma aplicação e são gerenciados por uma única entidade. Em IdC, por outro lado, os dispositivos executam diferentes aplicações e respondem a mais de uma autoridade. Nossa solução AdC provê autenticação e controle de acesso para IdC, atendendo aos requisitos e restrições específicos de IdC.

Autenticação e controle de acesso para IdC. Na literatura, os trabalhos que abordam autenticação e controle de acesso sob uma perspectiva de IdC propõe diferentes estratégias, *e.g.,* através de políticas de gerenciamento, verificação de dados, aplicação de CBI, tecnologias específicas para pareamento de dispositivos, definição de padrões de segurança, entre outros. A seguir descrevemos brevemente alguns trabalhos que dão uma visão geral do estado da arte.

Recentemente, [Liang et al. 2015] apresentaram *Safe Internet oF Things* – SIFT, uma plataforma de programação centrada em segurança para dispositivos conectados em ambientes de IdC. No SIFT, os usuários expressam suas intenções de alto nível em

aplicações de IdC e o esquema, então, decide quais dados e operações devem ser combinados para atender às necessidades do usuário. Para garantir segurança e conformidade, o esquema verifica se podem ocorrer conflitos ou violações de políticas dentro ou entre as aplicações, o que pode ser considerado como um mecanismo de controle de acesso. Estritamente falando, no entanto, tal trabalho tem foco na segurança dos usuários.

O trabalho de [Oliveira et al. 2009], por sua vez, aborda a autenticação da comunicação entre um dispositivo e múltiplos usuários. A proposta, chamada *Secure Tiny Web Services – Secure-TWS*, reporta a sobrecarga causada por estratégias de assinatura digital, mais especificamente, pelo esquema de Assinatura Digital Baseada em Curvas Elípticas (*Elliptic Curve Digital Signature Algorithm* – ECDSA) e o esquema de assinaturas curtas de Boneh-Lynn-Shacham (BLS).

[Mora-Afonso et al. 2013] propõe um esquema para fornecer segurança na comunicação de dispositivos domésticos. Os autores empregam CBI e *Wi-Fi*, assim como *Bluetooth* e NFC para autenticar as mensagens trocadas pelos dispositivos. O NFC é usado durante o pareamento por *Bluetooth* e a distribuição de chaves. Além disto, os autores apresentam protótipos para diversos dispositivos.

[Markmann et al. 2015] propõe um esquema de autenticação fim a fim para implantar uma federação de *gateways* em sub-redes de IdC. A estratégia proposta é, também, baseada em CBI, e, apesar de preliminar, traz contribuições significativas para a área.

Por fim, [Yavuz 2013] desenvolveu um esquema criptográfico destinado a dispositivos IdC chamado *Efficient and Tiny Authentication* – ETA. ETA é similar ao AdC no sentido de suportar autenticação e controle de acesso, entretanto, não considera todo o ciclo de vida dos dispositivos de IdC e, além disso, é baseada em dispositivos com abundância de recursos computacionais para executar operações custosas.

6. Conclusão

À medida que IdC se torna ubíqua, a troca de informações privadas, assim como a execução automática de operações nos ambientes computacionais se tornam fator crítico. Nestes cenários é imperativo que a comunicação seja autenticada e que haja meios seguros de controle de acesso às operações dos dispositivos. Este trabalho propõe AdC, uma família de protocolos que provê autenticação e controle de acesso a todo o ciclo de vida de dispositivos de IdC. O AdC possibilita autenticação forte e controle de acesso flexível através da combinação de primitivas criptográficas do estado da arte.

O AdC foi avaliado analítica e experimentalmente. Os resultados analíticos mostram que a sobrecarga imposta pelo AdC tanto para CPU quanto para comunicação, no pior caso, são gerenciáveis até mesmo em dispositivos com limitações de recursos. Nós também utilizamos uma ferramenta de verificação automática para validar as propriedades de segurança do AdC. Por fim, os resultados experimentais mostram que o AdC pode ser executado com baixo custo em dispositivos com restrições computacionais e com custo desprezível em dispositivos computacionalmente poderosos.

Referências

Aranha, D. F. and Gouvêa, C. P. L. RELIC is an Efficient LIbrary for Cryptography. `https://github.com/relic-toolkit/relic`.

Ashton, K. (2009). That 'Internet of Things' Thing. *RFiD Journal*, 22:97–114.

Barreto, P. S. L. M. and Naehrig, M. (2005). Pairing-friendly Elliptic Curves of Prime Order. In *SAC*.

Bethencourt, J., Sahai, A., and Waters, B. (2007). Ciphertext-policy Attribute-based Encryption. In *S&P*.

Boneh, D. and Franklin, M. K. (2001). Identity-Based Encryption from the Weil Pairing. In *CRYPTO*.

Cao, X., Kou, W., Dang, L., and Zhao, B. (2008). IMBAS: Identity-based Multi-user Broadcast Authentication in Wireless Sensor Networks. *Computer Communications*, 31(4):659 – 667.

Cremers, C. (2008). The scyther tool: Verification, falsification, and analysis of security protocols. In *CAV*.

Goyal, V., Pandey, O., Sahai, A., and Waters, B. (2006). Attribute-Based Encryption for Fine-Grained Access Control of Encrypted Data. In *CCS*.

Liang, C.-J. M., Karlsson, B. F., Lane, N. D., Zhao, F., Zhang, J., Pan, Z., Li, Z., and Yu, Y. (2015). SIFT: Building an Internet of Safe Things. In *IPSN*.

Liu, D., Ning, P., and Li, R. (2005). Establishing Pairwise Keys in Distributed Sensor Networks. *TISSEC*.

Maji, H. K., Prabhakaran, M., and Rosulek, M. (2008). Attribute-Based Signatures: Achieving Attribute-Privacy and Collusion-Resistance. *IACR Cryptology ePrint Archive*, 2008:328.

Margi, C., Simplicio, M., Barreto, P., and Carvalho, T. (2009). Segurança em redes de sensores sem fio. *Minicursos: SBSeg*.

Markmann, T., Schmidt, T. C., and Wählisch, M. (2015). Federated End-to-End Authentication for the Constrained Internet of Things Using IBC and ECC. In *SIGCOMM*.

McCullagh, N. and Barreto, P. S. L. M. (2005). A New Two-party Identity-based Authenticated Key Agreement. In *CT-RSA*.

Mora-Afonso, V., Caballero-Gil, P., and Molina-Gil, J. (2013). Strong Authentication on Smart Wireless Devices. In *FGTC*.

Oliveira, L. B. and Dahab, R. (2006). Pairing-based cryptography for sensor networks. In *NCA*.

Oliveira, L. B., Kansal, A., Priyantha, B., Goraczko, M., and Zhao, F. (2009). Secure-TWS: Authenticating Node to Multi-user Communication in Shared Sensor Networks. In *IPSN*.

Oliveira, L. B., Scott, M., Lopez, J., and Dahab, R. (2008). TinyPBC: Pairings for Authenticated Identity-Based Non-Interactive Key Distribution in Sensor Networks. In *INSS*.

Patil, H. K. and Szygenda, S. A. (2012). *Security for Wireless Sensor Networks Using Identity-Based Cryptography*. CRC Press.

Perrig, A., Szewczyk, R., Wen, V., Culler, D. E., and Tygar, J. D. (2001). SPINS: Security Protocols for Sensor Netowrks. In *MobiCom*.

Sakai, R., Ohgishi, K., and Kasahara, M. (2000). Cryptosystems Based on Pairing. In *SCIS*.

Shamir, A. (1984). Identity-based Cryptosystems and Signature Schemes. In *CRYPTO*.

Silva, B., da Silva Jr, D. C., Souza, E. M., Pereira, F., Teixeira, F., Wong, H. C., Nazaré, H., Maffra, I., Freire, J., Santos, W. F., et al. (2013). Segurança de software em sistemas embarcados: Ataques & defesas. *Minicursos: SBSeg*.

Stinson, D. (2002). *Cryptography: Theory and Practice*. CRC/C&H.

Venkatraman, L. and Agrawal, D. P. (2002). A novel authentication scheme for ad hoc networks. In *WCNC*.

Wangham, M. S., Domenech, M. C., and de Mello, E. R. (2013). Infraestrutura de autenticação e de autorização para internet das coisas. *Minicursos: SBSeg*.

Yavuz, A. A. (2013). ETA: Efficient and Tiny and Authentication for Heterogeneous Wireless Systems. In *WiSec*.

Yuan, E. and Tong, J. (2005). Attributed Based Access Control (ABAC) for Web Services. In *ICWS*.

Zhou, L. and Haas, Z. J. (1999). Securing Ad Hoc Networks. *IEEE Network*, 13(6):24–30.

Zhu, S., Setia, S., and Jajodia, S. (2003). LEAP: Efficient Security Mechanisms for Large-scale Distributed Sensor Networks. In *CCS*.

