Autenticação e comunicação segura em dispositivos móveis de poder computacional restrito

Rafael Will Macedo de Araujo[1], **Routo Terada**[1]

[1]Instituto de Matemática e Estatística – Universidade de São Paulo (USP)
Rua do Matão, 1010 - Butantã – São Paulo – SP – Brazil

{rwill,rt}@ime.usp.br

Abstract. *Protocols for authentication and key establishment are fundamental parts in security implementations for electronic devices communication. In applications involving devices with limited computational power communicating with a server, the choice of efficient protocols that require a simpler infrastructure is essential. In this work we implement secure key agreement protocols in ID-based and Certificateless public key cryptography models on ARM processor platforms. We also compare running times, memory and network usage.*

Resumo. *Protocolos de autenticação e de estabelecimento de chaves são peças fundamentais em implementações de segurança para comunicação de dispositivos eletrônicos. Em aplicações que envolvam dispositivos com poder computacional restrito comunicando-se com um servidor, é fundamental a escolha de protocolos eficientes e que necessitem de uma infraestrutura mais simples. Neste trabalho implementamos protocolos de acordo de chave seguros nos modelos de criptografia de chave pública baseado em identidade (ID-based) e sem certificado (Certificateless) em plataformas com processadores ARM. Comparamos tempos de execução, utilização de memória e uso do canal de comunicação.*

1. Introdução

O gerenciamento de certificados digitais requer uma infraestrutura de suporte mais robusta, o que acarreta em maiores custos de implementação e manutenção. Ademais, há uma série de dificuldades nos processos de de recuperação, validação e revogação de certificados. De maneira geral, os sistemas convencionais utilizam o criptossistema RSA, baseado no problema de fatoração de números inteiros grandes. Neste tipo de sistema as chaves públicas são números escolhidos aleatoriamente, e não possuem qualquer vínculo com o seu dono.

Os modelos de criptografia de chave pública baseado em identidade e sem certificado são alternativas ao modelo convencional de criptografia assimétrica. Em ambos os modelos a autenticação da chave pública ocorre implicitamente, durante a execução dos protocolos, sem a necessidade de gerenciar e distribuir certificados digitais. Tais sistemas, aliados ao uso de curvas elípticas, acarretam em menor custo operacional e computacional.

Neste contexto, o uso de criptografia de curvas elípticas se torna muito atraente para ambientes que dispõem menor capacidade de processamento. Através do uso de

*Este trabalho foi financiado pelo CNPq

*Dissertação de mestrado disponível em: http://www.ime.usp.br/~rwill/dissertacao

protocolos de acordo de chave com autenticação, é possível garantir confidencialidade, integridade e autenticidade de usuários e de equipamentos, sem prévia distribuição de segredos compartilhados, e estabelecer chaves compartilhadas entre dois usuários remotos, para uso posterior com algoritmos de criptografia simétrica.

2. Motivação e Contribuição

Dado um cenário onde um dispositivo com poder baixo poder computacional precisa se comunicar com um servidor, implementamos, de maneira eficiente, protocolos de acordo de chave nos modelos de criptografia baseado em identidade (*ID-Based*) [Shamir 1984], e sem certificado (*Certificateless*) [Al-Riyami and Paterson 2003]. Mostramos que é viável utilizar protocolos relativamente complexos, com níveis elevados de segurança, e tempos de execução praticáveis. Nossos testes simularam situações do mundo real, através de comunicação em uma rede TCP/IP, e utilização de bibliotecas criptográficas e matemáticas otimizadas, construídas principalmente em linguagens C e Assembly.

3. Problemas computacionais

A segurança dos protocolos analisados neste trabalho recai sobre quatro problemas computacionais, derivados do problema do logaritmo discreto em curvas elípticas. São eles:

- **DDH** (*Decision Diffie-Hellman Problem*): dados: $P, aP, bP, cP \in \mathbb{G}$; decidir: $abP = cP$.
- **CDH** (*Computacional Diffie-Hellman Problem*): dados: $P, aP, bP \in \mathbb{G}$; encontrar: abP.
- **GBDH** (*Gap-Bilinear Diffie-Hellman Problem*): dados: $P, aP, bP \in \mathbb{G}$; encontrar: abP, com ajuda de um oráculo de decisão (que dados $aP, bP, cP \in \mathbb{G}$, decide se $abP = cP$).
- **BDH** (*Bilinear Diffie-Hellman Problem*): dados: $P, aP, bP, cP \in \mathbb{G}$; encontrar: $e(P, P)^{abc}$.

Do ponto de vista teórico, um protocolo que tenha sido demonstrado seguro sob a hipótese de dificuldade do problema BDH é, pelo menos, tão seguro quanto outro que tenha sido demonstrado seguro sob a hipótese de dificuldade do problema Gap-BDH, no mesmo modelo de segurança. Por esse motivo, é preferível utilizar protocolos que se apoiem na suposição de dificuldade do problema BDH, e o uso do Gap-BDH deve ser evitado sempre que possível [Goya 2011].

4. Protocolos Analisados

A escolha dos protocolos a serem analisados levou em consideração os seguintes aspectos: modelo de segurança empregado, relevância do protocolo (quantidade de citações aos artigos em que são apresentados), ano de publicação (protocolos apresentados recentemente) e ausência de publicações científicas que comprovem falha de segurança ou quebra do protocolo. Outra exigência crucial foi a garantia de autenticidade dos usuários (isto é, protocolos de acordo de chave com autenticação, ou AKA - *Authenticated Key Agreement*).

Os protocolos escolhidos no modelo baseado em identidade foram demonstrados seguros sobre os modelos CK e eCK, definidos em [Canetti and Krawczyk 2001]

e [LaMacchia et al. 2007] respectivamente. Já protocolos escolhidos no modelo sem certificado foram demonstrados seguros sobre os modelos LBG (definido em [Lippold et al. 2009]), SJ$^+$, Mal-LBG e Mal-SJ$^+$ (definidos em [Goya 2011]). Todos são extensões do modelo eCK que incluem adaptações para o caso sem certificado. Na sequência, são listados os protocolos de acordo de chave escolhidos, o problema computacional e o modelo de segurança no qual se baseiam. Maiores detalhes sobre a descrição dos protocolos e suas operações estão disponíveis em [Araujo 2013].

Protocolos no modelo baseado em identidade:

- **HC-BDH**: seguro no modelo eCK, [Huang and Cao 2009]
- **HLZ-GBDH**: seguro no modelo eCK, [Hu et al. 2009]
- **CC-BDH**: seguro no modelo CK, [Chow and Choo 2007]
- **NCLH-BDH** e **NCLH-GBDH**: seguro no modelo eCK, [Ni et al. 2011]
- **NCL-BDH**: seguro no modelo eCK, [Ni et al. 2012]

Protocolos no modelo sem certificado:

- **LBG-BDH** e **LBG-GBDH**: seguro no modelo LBG, [Lippold et al. 2009]
- **GOT-BDH** e **GOT-GBDH**: seguro no modelo LBG, [Goya et al. 2010]
- **GNT3-BDH**: seguro no modelo LBG, [Goya 2011]
- **GNT1-GBDH**: seguro no modelo Mal-LBG, [Goya et al. 2011]
- **GNT2-GBDH**: seguro no modelo Mal-SJ$^+$, [Goya 2011]
- **GNT4-BDH**: seguro no modelo SJ$^+$, [Goya 2011]

5. Experimentos

O cenário proposto para os experimentos consiste em acordar uma chave secreta entre um dispositivo de baixo poder computacional (cliente) e um servidor, que funciona como a autoridade do sistema, conforme ilustrado na Figura 1. Neste ambiente não há distribuição prévia de segredos compartilhados. A comunicação de ambos acontece por meio de uma rede TCP/IP, utilizando os padrões IEEE 802.11g (para comunicação sem fio) e 100BASE-TX (*Fast Ethernet*), a depender do dispositivo.

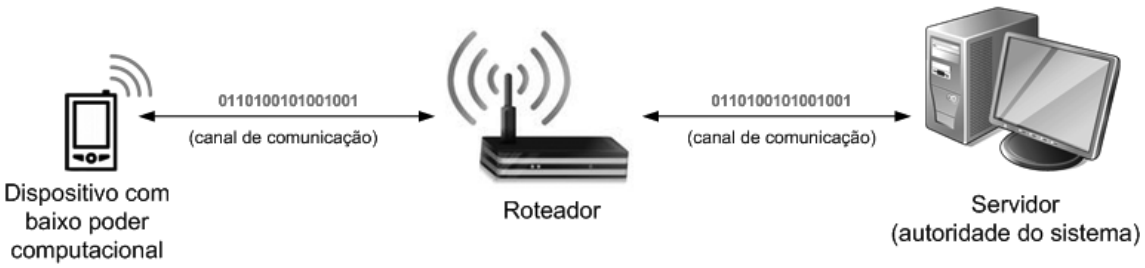

Figura 1. Cenário de comunicação entre o dispositivo e o servidor (autoridade do sistema) [Araujo and Terada 2013]

Os experimentos foram realizados utilizando-se três dispositivos como clientes: um *smartphone*, um *tablet*, e um *single-board computer* voltado para sistemas embarcados. Tais dispositivos dispõem de sistema operacional Android ou Linux. A seguir são detalhadas as configurações dos equipamentos utilizados:

- **Servidor (PC)**: processador Intel Core 2 Duo T5800 de 2.0 GHz, memória RAM de 3 GB, e sistema operacional Ubuntu 12.04 LTS de 32 bits.

- **Smartphone (MM1)**: Motorola Milestone 1, processador ARM Cortex-A8 (*single-core*) de 600 MHz, memória RAM de 256 MB, e sistema operacional Android 2.2.
- **Tablet (GN7)**: Asus Google Nexus 7, processador ARM Cortex-A9 (*quad-core*) de 1.2 GHz, memória RAM de 1 GB, e sistema operacional Android 4.2.
- **Single-board computer (RPi)**: Raspberry Pi, processador ARM1176JZF-S (*single-core*) de 700 MHz, memória RAM de 256 MB, e sistema operacional Debian "Wheezy" (armhf).

5.1. Ambiente de desenvolvimento

Escolhemos a *RELIC-toolkit* (versão 0.3.1) [Aranha and Gouvêa 2009] como biblioteca criptográfica, construída em linguagens C e Assembly. Consequentemente, as implementações dos protocolos estudados foram escritas em linguagem C, integrando-se de forma transparente à biblioteca *RELIC-toolkit*. A comunicação entre os dispositivos e o servidor é realizada através de *sockets* em C. Dentre as vantagens da utilização de *sockets* estão a fácil integração dos códigos construídos com as funções de comunicação em rede, comunicação TCP/IP direta, e facilidade em transmitir alguns tipos de dados complexos da *RELIC-toolkit*, já que não existem funções nativas que convertam tais tipos para *strings* e vice-versa.

5.2. Curvas elípticas e custo das operações

A biblioteca *RELIC-toolkit* dispõe de um conjunto de curvas elípticas pré-definidas. Tais curvas se dividem em dois grupos: curvas elípticas sobre corpos binários e curvas elípticas sobre corpos primos. Nos interessam as curvas conhecidas como *pairing-friendly*, que permitem a construção de sistemas criptográficos que fazem uso de emparelhamentos bilineares [Freeman et al. 2010].

Foram selecionadas seis curvas elípticas que oferecem diferentes níveis de segurança, sendo três definidas sobre corpos binários e três definidas sobre corpos primos. As curvas binárias escolhidas possuem grau de mergulho 4, e estão definidas sobre corpos binários de 271, 353 e 1223 bits. Estas curvas trabalham com emparelhamento simétrico. No caso das curvas primas, foram selecionadas curvas elípticas conhecidas como BN [Barreto and Naehrig 2006], definidas sobre corpos primos de 158, 256 e 638 bits. Estas curvas trabalham com emparelhamento assimétrico e possuem grau de mergulho 12, permitindo assim o uso de corpos algébricos menores, contudo, mantendo elevado nível de segurança. Por este motivo, são mais eficientes quando comparadas com curvas binárias.

Para mensurar o desempenho de cada uma dessas curvas, medimos o tempo de execução das principais operações envolvidas nos protocolos analisados, conforme a Tabela 1. Para curvas primas, faz-se necessário mensurar operações em $\mathbb{G}_1$ e em $\mathbb{G}_2$, já que $\mathbb{G}_1 \neq \mathbb{G}_2$. Foram utilizadas 50 amostras de tempo para o cálculo de cada um dos valores apresentados.

Tabela 1. Intervalo de confiança (95%) dos tempos de execução (em milissegundos) das operações para diferentes níveis de segurança

		Nível de segurança das curvas elípticas binárias		
Operação	**Dispositivo**	**70 bits**	**80 bits**	**128 bits**
Emparelhamento ηT	PC	[5.14, 5.15]	[8.61, 8.61]	[117.0, 117.15]
	RPi	[45.46, 45.57]	[79.46, 79.61]	[1229.19, 1229.56]
	MM1	[42.16, 42.56]	[72.9, 73.43]	[1275.74, 1295.76]
	GN7	[16.44, 16.48]	[51.87, 51.97]	[479.47, 479.98]
Exponenciação em $\mathbb{G}_T$	PC	[5.93, 6.06]	[10.63, 10.81]	[149.97, 151.74]
	RPi	[52.45, 53.56]	[98.86, 100.48]	[1572.75, 1591.09
	MM1	[50.06, 51.2]	[93.94, 95.85]	[1641.15, 1666.62]
	GN7	[19.45, 19.89]	[66.42, 68.12]	[613.84, 621.1]
Multiplicação em $\mathbb{G}_T$	PC	[0.04, 0.04]	[0.06, 0.06]	[0.23, 0.23]
	RPi	[0.38, 0.39]	[0.52, 0.53]	[2.42, 2.47]
	MM1	[0.35, 0.36]	[0.47, 0.51]	[2.45, 2.97]
	GN7	[0.14, 0.14]	[0.33, 0.34]	[0.96, 0.96]
Multiplicação em $\mathbb{G}$	PC	[3.44, 3.48]	[6.09, 6.15]	[75.05, 75.53]
	RPi	[30.81, 31.15]	[56.0, 56.57]	[781.48, 785.98]
	MM1	[29.38, 31.82]	[53.65, 54.42]	[828.44, 838.29]
	GN7	[11.5, 11.63]	[37.73, 38.26]	[298.53, 300.18]
Adição em $\mathbb{G}$	PC	[0.03, 0.03]	[0.03, 0.04]	[0.13, 0.13]
	RPi	[0.25, 0.26]	[0.32, 0.33]	[1.43, 1.45]
	MM1	[0.22, 0.23]	[0.29, 0.3]	[1.44, 1.9]
	GN7	[0.09, 0.09]	[0.21, 0.21]	[0.56, 0.56]

		Nível de segurança das curvas elípticas primas		
Operação	**Dispositivo**	**78 bits**	**128 bits**	**192 bits**
Emparelhamento *Optimal Ate*	PC	[5.47, 5.48]	[13.07, 13.08]	[90.06, 90.08]
	RPi	[42.22, 42.33]	[94.66, 94.83]	[624.75, 625.04]
	MM1	[49.34, 50.67]	[110.67, 112.57]	[684.97, 695.99]
	GN7	[31.37, 31.92]	[68.14, 68.19]	[408.1, 409.0]
Exponenciação em $\mathbb{G}_T$	PC	[5.27, 5.35]	[12.59, 12.74]	[88.07, 88.74]
	RPi	[40.88, 41.54]	[91.04, 92.17]	[612.32, 616.96]
	MM1	[48.15, 49.64]	[107.83, 110.25]	[673.39, 685.02]
	GN7	[30.49, 31.25]	[66.42, 67.27]	[401.86, 405.3]
Multiplicação em $\mathbb{G}_T$	PC	[0.04, 0.04]	[0.05, 0.05]	[0.15, 0.15]
	RPi	[0.27, 0.27]	[0.37, 0.38]	[1.02, 1.06]
	MM1	[0.27, 0.27]	[0.36, 0.64]	[0.98, 1.24]
	GN7	[0.18, 0.19]	[0.25, 0.26]	[0.66, 0.66]
Multiplicação em $\mathbb{G}_1$	PC	[0.56, 0.56]	[1.4, 1.4]	[10.01, 10.06]
	RPi	[4.5, 4.54]	[10.14, 10.2]	[70.32, 70.65]
	MM1	[4.91, 5.46]	[11.03, 11.75]	[77.9, 79.63]
	GN7	[3.5, 3.57]	[7.55, 7.6]	[45.6, 46.47]
Multiplicação em $\mathbb{G}_2$	PC	[2.35, 2.38]	[6.07, 6.13]	[47.31, 47.62]
	RPi	[16.98, 17.22]	[41.25, 41.7]	[316.97, 319.02]
	MM1	[19.78, 20.78]	[48.25, 51.73]	[348.66, 354.51]
	GN7	[12.77, 13.06]	[29.57, 29.89]	[206.17, 207.75]
Adição em $\mathbb{G}_1$	PC	[0.004, 0.004]	[0.01, 0.01]	[0.02, 0.02]
	RPi	[0.03, 0.03]	[0.04, 0.05]	[0.12, 0.12]
	MM1	[0.04, 0.05]	[0.05, 0.05]	[0.09, 0.32]
	GN7	[0.02, 0.02]	[0.03, 0.03]	[0.07, 0.08]
Adição em $\mathbb{G}_2$	PC	[0.01, 0.01]	[0.01, 0.01]	[0.04, 0.04]
	RPi	[0.07, 0.07]	[0.1, 0.1]	[0.26, 0.28]
	MM1	[0.07, 0.07]	[0.1, 0.11]	[0.23, 0.41]
	GN7	[0.05, 0.05]	[0.06, 0.06]	[0.17, 0.17]

Utilizamos a biblioteca GMP (GNU *Multiple Precision Library*) como *backend* aritmético da *RELIC-toolkit*. Desta forma, o tempo médio das operações diminuiu na razão de 3 a 8 vezes, dependendo do dispositivo e do nível de segurança. Pode-se observar, pela Tabela 1, que as operações que demandam mais tempo são emparelhamento, exponenciação em $\mathbb{G}_T$ (que em curvas binárias chega a ser mais custosa que emparelhamento), multiplicação escalar nos grupos $\mathbb{G}$ (curvas binárias) e em $\mathbb{G}_2$ e $\mathbb{G}_1$ (curvas primas).

Observa-se que o desempenho das curvas primas foi muito superior ao das curvas binárias, mesmo as primeiras apresentando um nível de segurança mais elevado. Devido à existência de dois grupos algébricos nos corpos de característica prima, podemos utilizar a propriedade de bilinearidade dos emparelhamentos para converter operações de exponenciação no grupo $\mathbb{G}_T$ em multiplicações nos grupos $\mathbb{G}_1$ ou $\mathbb{G}_2$, resultando em um ganho de eficiência maior que 50% para este tipo de conversão.

Outra observação relevante é que o problema do logaritmo discreto em corpos de característica pequena (que é o caso das curvas binárias utilizadas) foi criptoanalisado recentemente em [Barbulescu et al. 2014]. Desta maneira, além de menos eficientes que as curvas primas, o uso de curvas elípticas binárias é considerado inseguro atualmente.

5.3. Adaptação eficiente para emparelhamento assimétrico

Uma forma de implementar protocolos mais eficientes, ao utilizar curvas primas, é distribuir os valores dos grupos $\mathbb{G}_1$ e $\mathbb{G}_2$ de forma que o dispositivo com menor poder computacional (cliente) realize mais cálculos no grupo $\mathbb{G}_1$, enquanto o servidor (que possui maior poder computacional) realize mais operações no grupo $\mathbb{G}_2$. Como visto na Tabela 1, operações em $\mathbb{G}_2$ chegam a custar em média quatro vezes mais que operações em $\mathbb{G}_1$, e por este motivo devemos maximizar a quantidade de multiplicações em $\mathbb{G}_1$ no lado do cliente.

Em [Araujo and Terada 2013] foram apresentadas duas adaptações do protocolo Huang-Cao [Huang and Cao 2009] para emparelhamento assimétrico. Na primeira delas, o servidor terá que executar duas multiplicações em $\mathbb{G}_1$ e duas multiplicações em $\mathbb{G}_2$, e o cliente executará três multiplicações em $\mathbb{G}_2$, que são mais custosas. Já o segundo cenário é mais vantajoso para o cliente, que calculará apenas três multiplicações em $\mathbb{G}_1$, e o servidor continua com duas multiplicações em $\mathbb{G}_1$ e duas multiplicações em $\mathbb{G}_2$. Adaptações similares a esta foram produzidas para todos os protocolos implementados, com o objetivo de tornar o código utilizado pelos dispositivos o mais eficiente possível.

5.4. Desempenho dos protocolos analisados

Em [Araujo 2013] e [Araujo and Terada 2013] são apresentadas comparações entre os tempos de execução dos protocolos estudados, nos três dispositivos analisados, para as seis curvas elípticas selecionadas. A contagem de tempo iniciou-se no momento da escolha dos valores temporários, e encerrou-se após o cálculo da chave de sessão. Para cada protocolo, em cada dispositivo, foram obtidas um total de 100 amostras de tempo, sendo 50 para o caso sem pré-computação, e 50 para o caso com pré-computação. Para cada conjunto de 50 amostras calculou-se a média e o intervalo de confiança (de 95%), sendo que cada amostra corresponde a uma sessão completa, ou seja, escolhem-se novos valores temporários e calcula-se uma nova chave de sessão, diferente das anteriores. O cenário sem pré-computação representa a execução dos protocolos da maneira como são definidos. Já o cenário com pré-computação considera que uma entidade A se comunica frequentemente com outra entidade B e, desta forma, alguns valores referentes a B não precisam ser recalculados a cada nova sessão.

Os resultados confirmam que protocolos sob modelos de segurança mais fortes e problemas computacionais mais difíceis são mais lentos. Alguns protocolos são versões aprimoradas de outros um pouco mais antigos, como é o caso do NCLH-BDH e NCL-BDH, e dos protocolos GNT3, GOT e LBG. Nestes casos houve melhoria de eficiência ou

equivalência, sendo que nos casos onde o tempo é equivalente o consumo de memória foi menor. Também em [Araujo 2013] e [Araujo and Terada 2013] são apresentadas tabelas que comparam a quantidade de memória RAM utilizada e o comprimento das mensagens trocadas (ambos em *bytes*) por protocolo e curva elíptica.

6. Conclusão

Este trabalho propôs implementações eficientes de protocolos de acordo de chave em dispositivos de poder computacional restrito nos modelos de criptografia baseado em identidade e sem certificado. Os resultados dos experimentos mostram que é possível utilizar tais protocolos com tempos praticáveis. Apontamos sugestões para desenvolvedores de como implementar protocolos de forma eficiente, através da redistribuição das operações de multiplicação nos grupos $\mathbb{G}_1$ e $\mathbb{G}_2$, quando utilizado emparelhamento assimétrico, ou substituição de operações de exponenciação no grupo $\mathbb{G}_T$ por multiplicações em $\mathbb{G}$ quando a troca é de um para um. Também verificou-se o uso de memória RAM e comprimento das mensagens trocadas que, juntamente com as comparações de tempo, formam um guia para desenvolvedores que pretendam utilizar algum desses protocolos em situações reais. As contribuições deste trabalho resultaram na publicação de [Araujo 2013].

Referências

Al-Riyami, S. S. and Paterson, K. G. (2003). Certificateless public key cryptography. In *Advances in Cryptology - ASIACRYPT 2003*, volume 2894 of *Lecture Notes in Computer Science*, Taipei, Taiwan. Springer.

Aranha, D. F. and Gouvêa, C. P. L. (2009). RELIC is an Efficient LIbrary for Cryptography. `http://code.google.com/p/relic-toolkit/`.

Araujo, R. W. M. (2013). Autenticação e comunicação segura em dispositivos móveis de poder computacional restrito. Master's thesis, Instituto de Matemática e Estatística da Universidade de São Paulo.

Araujo, R. W. M. and Terada, R. (2013). Implementação eficiente de protocolos de acordo de chave em dispositivos de poder computacional restrito. *SBSeg 2013, XIII Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais*, pages 323–336.

Barbulescu, R., Gaudry, P., Joux, A., and Thomé, E. (2014). A heuristic quasi-polynomial algorithm for discrete logarithm in finite fields of small characteristic. In *Advances in Cryptology–EUROCRYPT 2014*, pages 1–16. Springer.

Barreto, P. S. and Naehrig, M. (2006). Pairing-friendly elliptic curves of prime order. In *Selected areas in cryptography*, pages 319–331. Springer.

Canetti, R. and Krawczyk, H. (2001). Analysis of key-exchange protocols and their use for building secure channels. In *Advances in Cryptology-EUROCRYPT 2001*, pages 453–474. Springer.

Chow, S. S. and Choo, K.-K. R. (2007). Strongly-secure identity-based key agreement and anonymous extension. In *Information Security*, pages 203–220. Springer.

Freeman, D., Scott, M., and Teske, E. (2010). A taxonomy of pairing-friendly elliptic curves. *Journal of Cryptology*, 23(2):224–280.

Goya, D., Nakamura, D., and Terada, R. (2011). Acordo de chave seguro contra autoridade mal intencionada. *SBSeg 2011, XI Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais*, pages 265–278.

Goya, D., Okida, C., and Terada, R. (2010). A two-party certificateless authenticated key agreement protocol. *SBSeg 2010, X Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais*, pages 443–446.

Goya, D. H. (2011). *Criptografia de chave pública sem certificado*. PhD thesis, Instituto de Matemática e Estatística da Universidade de São Paulo.

Hu, X., Liu, W., and Zhang, J. (2009). An efficient id-based authenticated key exchange protocol. In *Information Engineering, 2009. ICIE'09. WASE International Conference on*, volume 2, pages 229–233. IEEE.

Huang, H. and Cao, Z. (2009). An id-based authenticated key exchange protocol based on bilinear diffie-hellman problem. In *Proceedings of the 4th International Symposium on Information, Computer, and Communications Security*, pages 333–342. ACM.

LaMacchia, B., Lauter, K., and Mityagin, A. (2007). Stronger security of authenticated key exchange. In *Provable Security*, pages 1–16. Springer.

Lippold, G., Boyd, C., and Gonzalez Nieto, J. (2009). Strongly secure certificateless key agreement. In *Proceedings of the 3rd International Conference Palo Alto on Pairing-Based Cryptography*, Pairing '09, pages 206–230, Berlin, Heidelberg. Springer-Verlag.

Ni, L., Chen, G., and Li, J. (2012). Escrowable identity-based authenticated key agreement protocol with strong security. *Computers & Mathematics with Applications*.

Ni, L., Chen, G., Li, J., and Hao, Y. (2011). Strongly secure identity-based authenticated key agreement protocols. *Computers & Electrical Engineering*, 37(2):205–217.

Shamir, A. (1984). Identity-based cryptosystems and signature schemes. In *Proceedings of CRYPTO 84 on Advances in cryptology*, volume LNCS 196, pages 47–53. Springer-Verlag New York, Inc.

