Capítulo 1

Introdução à Privacidade: Uma Abordagem Computacional

Fábio Borges
Laboratório Nacional de Computação Científica (LNCC)

Abstract

Information and communication technologies are continuously transforming how people interact in society. Many resources have been made available to simplify the lives of citizens. However, studies to ensure privacy have become increasingly needed. The leak of private information might have a serious impact on personal and professional life. The massive leak of private information might compromise the free will and democracy. The whole society can be manipulated. Considerable work has been done to protect privacy. This chapter aims to introduce techniques with their cryptographic primitives used to ensure privacy in various scenarios.

Resumo

Tecnologias da informação e comunicação estão continuamente transformando como as pessoas interagem na sociedade. Muitos recursos têm sido disponibilizados para simplificar a vida dos cidadãos. No entanto, estudos para garantir a privacidade têm se tornado cada vez mais necessários. O vazamento de uma informação privada pode causar sério impacto na vida pessoal e profissional. O vazamento massivo de informação privada pode comprometer o livre arbítrio e a democracia. Toda a sociedade pode ser manipulada. Muito tem sido feito para proteger a privacidade. Este capítulo visa introduzir técnicas com suas primitivas criptográficas usadas para garantir a privacidade em diversos cenários.

1.1. Introdução

Muitos estudantes que tiveram um curso de criptografia podem pensar que a privacidade depende apenas de uma decisão pessoal na qual ou se revela uma informação cifrada ou não se revela. No entanto, muitos problemas referentes à privacidade não são uma questão de escolha pessoal, mas são problemas computacionais complexos determinados por várias variáveis e oriundos de uma coletividade. Se por um lado, a garantia da privacidade está relacionada com problemas complexos, por outro lado o Direito à Privacidade é estabelecido no Artigo 12 da Declaração Universal dos Direitos Humanos (1948).

Este capítulo visa apresentar técnicas com suas primitivas criptográficas usadas para proteger a privacidade. Inicia-se apresentando diversos cenários nos quais a preservação da privacidade é fundamental. A violação do direito à privacidade em alguns destes cenários pode até mesmo comprometer a democracia. As seções subsequentes apresentam técnicas que são independentes de cenários específicos. Neste sentido, este texto é teórico, apesar de ter concretas motivações de cunho prático. Além disso, este capítulo apresenta uma coleta dos principais resultados sobre privacidade e deixa para os alunos o conhecimento e as referências necessárias para se aprofundarem em algum tópico específico. As primitivas criptográficas usadas nas técnicas para proteger a privacidade serão introduzidas antes de serem utilizadas. Logo, um aluno de graduação em computação pode seguir este texto sem a necessidade de ter cursado uma disciplina de criptografia, nem de segurança, como pré-requisito. Ao final deste capítulo, além de estar informado sobre os problemas de privacidade, os alunos de graduação terão habilidade de tratá-los de forma mais consciente e aplicar as técnicas mais adequadas às características de um problema de privacidade.

1.1.1. Principais Objetivos

Os principais objetivos deste capítulo são conscientizar o aluno da necessidade de preservar a privacidade em alguns cenários, introduzir métricas para avaliar as técnicas de proteção da privacidade para que o aluno possa avaliar melhor as técnicas e seus resultados, introduzir tecnologias usadas para proteger a privacidade junto com o conhecimento matemático necessário para se entender as técnicas simétricas e assimétricas usadas na proteção da privacidade, e como último objetivo, apresentar uma comparação das técnicas ensinadas neste texto.

1.1.2. Escopo e Estrutura do Texto

Este capítulo apresenta uma seleção de protocolos com seus algoritmos clássicos, *i.e.*, criptografia baseada em mecânica quântica está fora do escopo deste trabalho. Também não se deseja varrer todas as técnicas criptográficas usadas para proteger a privacidade, mas se objetiva varrer os protocolos mais importantes para proteger a privacidade nas aplicações mencionadas na Seção 1.2. Na sequência, a Seção 1.3 apresenta as métricas para a privacidade. A Seção 1.4 apresenta as técnicas simétricas e outras tecnologias para proteger a privacidade. A Seção 1.5 apresenta as técnicas de comprometimento junto com técnicas assimetrias. A Seção 1.7 apresenta comparações entres as principais técnicas. A seção 6 finaliza o texto com as conclusões. Para simplificar a leitura, o apêndice contém listas de acrônimos, abreviações e símbolos, além de um pequeno glossário.

1.2. O que é a privacidade?

Já sabemos que é um dos direitos humanos, mas sua definição é dependente da cultura de uma sociedade e do cenário a que se aplica à privacidade. Por exemplo, em certos países, os cidadãos escrevem os nomes dos moradores na caixa de correio, noutros não. Pois em outros, o nome na caixa do correio pode representar uma ameaça à segurança e privacidade. Salário é algo completamente privado em alguns países, noutros é público.

Nesta seção encontramos casos onde a privacidade se faz necessária na grande maioria dos países. Esta seção visa apresentar os problemas relativos à privacidade em vários cenários, a saber, votação eletrônica, sistemas de reputação, redes de sensores, cibermedicina, processamento de imagem, dinheiro eletrônico, sensoriamento móvel, computação em múltiplas partes, mundo acadêmico, e redes inteligentes. Esta não é uma lista completa de cenário. Na leitura dos cenários, o leitor pode imaginar muitos outros cenários. Finaliza-se a seção com uma breve introdução às leis que asseguram o direito à privacidade ao redor do mundo.

1.2.1. Votação Eletrônica

Fraudes em eleições podem comprometer a democracia. Por isto, faz-se necessário que os sistemas computacionais possam garantir a segurança das mensagens nos processos eleitorais. A violação do direito de os eleitores votarem secretamente pode coagi-los a votarem em candidatos que não votariam. Eleitores coagidos podem alterar o resultado da eleição o que também compromete a democracia. Por isto, faz-se necessário que os sistemas computacionais possam garantir a privacidade dos eleitores nos processos eleitorais.

Os sistemas de votação eletrônica devem ser similares ao processo tradicional de votação [Gritzalis 2002]. No entanto, nem todas as eleições baseadas em processos tradicionais são corretas, mas os processos de votação eletrônica deveriam ser corretos. De qualquer forma, o voto deve ser secreto. Além disto, os sistemas devem permitir verificações [Gritzalis 2002], *e.g.*, se o voto foi incluído na conta e se esta foi calculada corretamente. Verificações podem causar conflitos com privacidade. Para evitar conflito, pode-se imprimir o voto e depositá-lo em uma urna física. Urnas eletrônicas poderiam ser verificadas comparando seus resultados com os totais das urnas físicas. [Gritzalis 2002] apresenta os requisitos e princípios para sistemas de votação eletrônica que são resumidos na Tabela 1.1.

Sistemas de votação eletrônica podem ser construídos com algoritmos criptográficos, *e.g.*, [Cramer et al. 1997] apresentam um sistema de criptografia homomórfica baseado em [El Gamal 1985]. Saindo da computação clássica, [Vaccaro et al. 2007] apresentam sistemas de votação baseados em mecânica quântica.

1.2.2. Sistemas de Reputação

Sistemas de reputação, sistemas de recomendação e modelos de confiança descrevem esquemas para um novo usuário fazer escolhas mais assertivas baseadas em experiências de outros usuários. Sistemas de reputação podem ser baseados em algoritmos de ordenação como o PageRank. Sistemas de recomendação podem ser semelhantes a processos eleitorais, *e.g.*, onde clientes votam na sua empresa preferida. Modelos de confiança tentam

Tabela 1.1: Requisitos e princípios para sistemas de votação.

Requisitos	Princípios
Generalidade	Isomorfo ao tradicional Elegibilidade
Liberdade	Não-coercivo Nenhuma propaganda no entorno (sem *boca de urna*) Capacidade de votação não-válida (nulos, brancos, *etc.*)
Igualdade	Igualdade de candidatos Igualdade de eleitores Um eleitor - um voto
Sigilo	Voto secreto Segurança *v.* transparência
Direto	Votação não monitorada, mas contada
Democracia	Confiança e transparência Verificabilidade e prestação de contas Confiança e segurança Simplicidade

medir o quanto se pode confiar em uma empresa desconhecida baseando em evidências ou experiências relatadas por outros usuários. Todas estas três áreas de pesquisa têm uma grande intersecção. Em particular, a privacidade dos clientes testemunhas deve ser mantida para que os esquemas possibilitem que o novo cliente faça uma escolha assertiva. [Jøsang et al. 2007] desenvolveram uma interessante pesquisa literária sobre este tema.

Figura 1.1 esquematiza os clientes testemunhas, que já tiveram interação com uma empresa, provendo recomendações a um novo cliente que poderá tomar uma decisão mais assertiva devido a experiências relatadas. Certamente, o novo cliente tem que confiar nos clientes testemunhas.

[Kerschbaum 2009] desenvolve um sistema de reputação baseado em criptografia homomórfica e recomenda o uso da técnica de criptografia homomórfica desenvolvida por [Paillier 1999].

1.2.3. Redes de Sensores

Redes de sensores são redes formadas por sensores autônomos distribuídos espacialmente para coletar informações físicas e ambientais, *e.g.*, temperatura, umidade, ruído, *etc.*

Falhas na segurança podem gerar prejuízos financeiros enquanto o adversário se beneficia, *e.g.*, erros em previsões meteorológicas levam agricultores a plantar e colher em momentos errados. Logo, a colheita será reduzida e o custo do produto elevado no mercado. Semelhantemente, falhas na privacidade também podem ter implicações financeiras. Informações privilegiadas podem determinar melhor a precificação de imóveis. Em particular, um adversário pode saber se uma casa está vazia, se tem um habitante, e outras informações dos habitantes. Para isto, ele apenas precisa comparar as medições internas com as externas a residência. [Chan and Perrig 2003] apresentam um resumo dos problemas de segurança e privacidade em redes de sensores. [Peter et al. 2010] apresen-

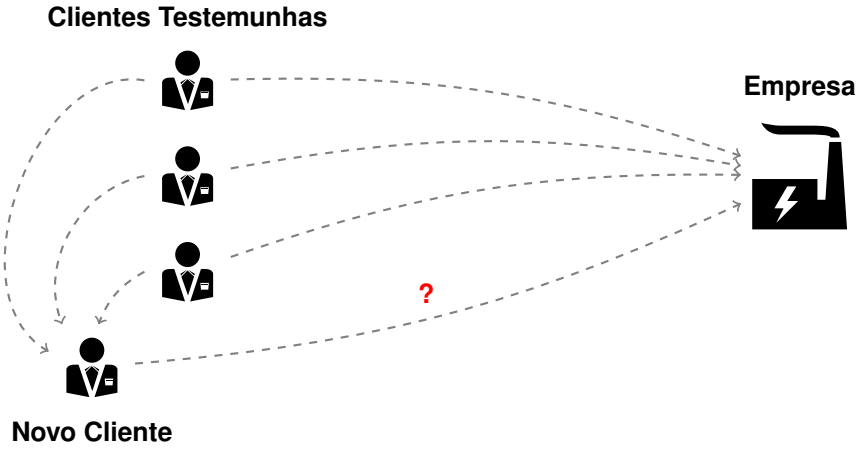

Figura 1.1: Cliente recebendo recomendações de outros clientes que já interagiram com uma empresa.

tam diversas técnicas de criptografia homomórfica para redes de sensores com especial atenção as baseadas em curvas elípticas.

Sensoriamento Móvel é um processo de coleta em dispositivos pessoais que são móveis formando um tipo especial de rede de sensores, e em geral, medem a posição geográfica de usuários de dispositivos móveis. [Li and Cao 2013] apresentam um esquema para agregar dados coletados, *e.g.*, a posição de vários usuários é agregada de forma que o agregador apenas descobre a quantidade de usuários em uma vizinhança.

1.2.4. Cibermedicina

Redes de sensores que coletam dados ligados a medicina também são sensíveis quanto a privacidade [Al Ameen et al. 2012]. Seguradoras e planos de saúde podem taxar clientes de forma diferenciada dependendo de dados coletados em tais redes. Um adversário poderia até mesmo saber dados sobre a emoção de um usuário de marca passos em determinados momentos.

Existem outros problemas de cibermedicina que não são relacionados a redes de sensores. Por exemplo, o prontuário eletrônico pode ser usado para rotular pessoas, e consequentemente, empresas poderiam se beneficiar ao violarem o sigilo médico. Em particular, dados de medicina do esporte poderiam privilegiar adversários.

[Bellare et al. 2007] apresentam como podemos fazer buscas em bases de dados criptografadas e [Naor and Shamir 1995] apresentam como podemos cifrar com várias chaves. Desta forma, poderíamos fazer pesquisas em prontuários sem ter acesso direto aos dados e poderíamos decifrar os prontuários com a chave do médico e do paciente, *i.e.*, quando ambos aprovassem o acesso.

1.2.5. Processamento de Imagem

[Zheng and Huang 2013] apresentam um esquema para proteção de imagens médicas baseado em criptografia homomórfica aditiva, *e.g.*, [Paillier 1999]. Além de imagens médicas, existem diversas aplicações de processamento de imagens que requerem cuidados

com a segurança e privacidade. Por exemplo, [Peter et al. 2013] apresentam um esquema para reconhecimento de faces que mantém a privacidade.

1.2.6. Dinheiro Eletrônico

As transações financeiras poderiam ser completamente eletrônicas. Entretanto, faz-se necessário manter a privacidade das transações. Caso contrário, cada pequena compra poderia ficar registrada para a vida toda. Com isto, um adversário poderia extrair informações de indivíduos e da sociedade que usar este recurso. Por exemplo, o adversário poderia saber sobre as escolhas, o comportamento, a personalidade, *etc.* Bitcoin é o dinheiro eletrônico mais conhecido. Uma análise de anonimato no uso de Bitcoins pode ser encontrada em [Reid and Harrigan 2013]. [Camenisch et al. 2007] apresentam um dos outros sistemas de dinheiro eletrônico. [Farhi et al. 2012] apresentam um sistema de dinheiro eletrônico baseado em mecânica quântica.

1.2.7. Computação em Múltiplas Partes

Computação em múltiplas partes é uma área de estudo que visa computar uma função em diversos dispositivos computacionais garantindo a segurança e privacidade dos dados. Em geral, a ideia é processar dados em nuvem de forma segura. Por exemplo, empresas de previsão meteorológica poderiam terceirizar o processamento sem preocupação de vazamento de informação. As buscas de novos medicamentos poderiam ser feitas em nuvem sem preocupações de vazamento de informação.

[Cramer et al. 2001] apresentam um esquema de computação em múltiplas partes baseada em um esquema de criptografia homomórfica aditiva criada por [Paillier 1999].

1.2.8. Privacidade no Mundo Acadêmico

Experimentos correlatos a seres humanos precisam passar por avaliações éticas. Além disto, existem outros problemas relacionados com privacidade. [Santini 2005] apresenta uma lista de dados confidenciais sobre revisões de importantes artigos científicos na área de computação que foram inicialmente rejeitados. Mais ainda, apresenta um link para um documento de avaliação de Albert Einstein com o nome de seu chefe que o avalia muito mal. Na academia, a privacidade é importante para não bloquear as pessoas de aprenderem por tentativa e erro. A privacidade dá o direito de avaliados e avaliadores errarem. Figura 1.2 ilustra que um revisor apresentou uma revisão classificando com nota contraria aos outros revisores gerando um ponto fora da curva, *i.e.*, um *outlier*. Os pontos em cinza no *boxplot* são as médias das avaliações. Certamente, os sistemas de avaliação devem classificar os melhores revisores e autores, mas isto deve ser feito na média dos últimos anos e não em pontos específicos.

Em instituições educacionais privadas onde se caracteriza uma relação de cliente e fornecedor, as leis dos direitos dos consumidores podem ser aplicadas.

> *Art. 43. O consumidor, sem prejuízo do disposto no art. 86, terá acesso às informações existentes em cadastros, fichas, registros e dados pessoais e de consumo arquivados sobre ele, bem como sobre as suas respectivas fontes.*
>
> *§1º Os cadastros e dados de consumidores devem ser objetivos, claros,*

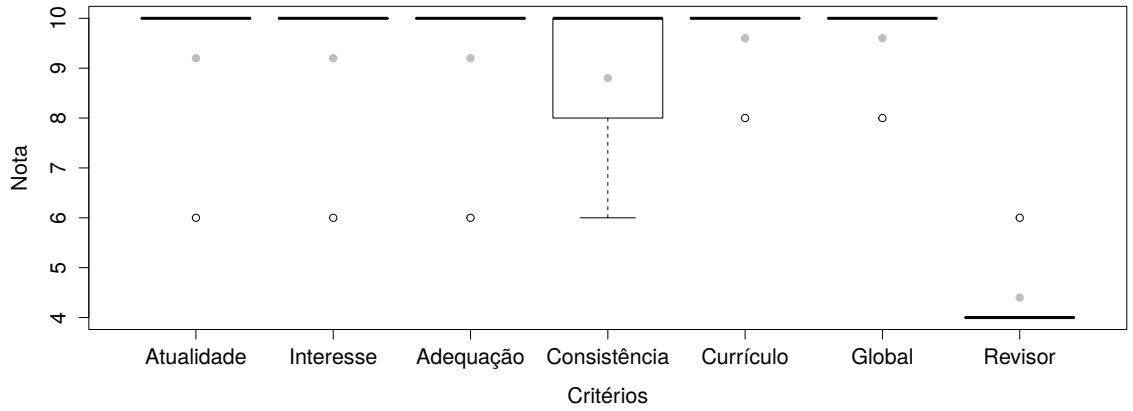

Figura 1.2: Nota obtida por critérios de avaliação.

> *verdadeiros e em linguagem de fácil compreensão, não podendo conter informações negativas referentes a período superior a* **cinco anos***.* [LEI Nº 8.078, DE 11 DE SETEMBRO DE 1990[1].]

> *Art. 59. Os órgãos públicos de defesa do consumidor devem providenciar a divulgação periódica dos cadastros atualizados de reclamações fundamentadas contra fornecedores.*
> *§3º Os cadastros deverão ser atualizados permanentemente, por meio das devidas anotações, não podendo conter informações negativas sobre fornecedores, referentes a período superior a* **cinco anos***, contado da data da intimação da decisão definitiva.* [DECRETO Nº 2.181, DE 20 DE MARÇO DE 1997[2].]

O tempo máximo de cinco anos deveria também ser usado na comunidade científica para retirada de informações sobre revisão de artigos científicos.

1.2.9. Redes Inteligentes (*Smart Grids*)

> *Smart grid é uma rede de pessoas, computadores, sensores em infraestruturas públicas que monitora e gerencia o uso de commodities.*
> [Borges de Oliveira 2017d]

Na maioria dos casos, os sensores são medidores inteligentes que enviam o consumo de eletricidade frequentemente para a companhia de energia. Neste caso, há diversos problemas de privacidade, *e.g.*, um adversário poderia detectar quando uma pessoa está em casa, se está sozinha, o que ela está assistindo na TV, que horas se levanta ou se deita, *etc.* [Borges de Oliveira 2017d]

[Borges de Oliveira 2017b] apresenta uma visão geral sobre redes inteligentes com os modelos de segurança e privacidade. [Borges de Oliveira 2017g] apresenta uma seleção de protocolos que protegem a privacidade dos consumidores.

[1] http://www.planalto.gov.br/ccivil_03/leis/L8078compilado.htm
[2] http://www.planalto.gov.br/ccivil_03/decreto/d2181.htm

Figura 1.3 esquematiza uma rede inteligente com os medidores inteligentes medindo o consumo da eletricidade e mandando os dados para suas respectivas companhias de distribuição de eletricidade. A figura apresenta duas redes, a saber, a rede de dados em azul e tracejado, e a rede elétrica em preto contínuo. Note que as medidas devem ser enviadas com frequência pela rede de dados as companhias, porque duas companhias estão virtualizando a rede de distribuição elétrica [Borges de Oliveira 2017f], *i.e.*, compartilhando uma rede de distribuição pública enquanto competem sem que uma terceira companhia controle o balanceamento das cargas elétricas. Logo, elas precisam prever o consumo com precisão para manterem equilíbrio da rede de distribuição elétrica. Para isso, precisam coletar e enviar os valores com máxima frequência, o que libera vários detalhes da vida privada dos clientes. Portanto, faz-se necessário o uso de protocolos que protejam a privacidade dos clientes.

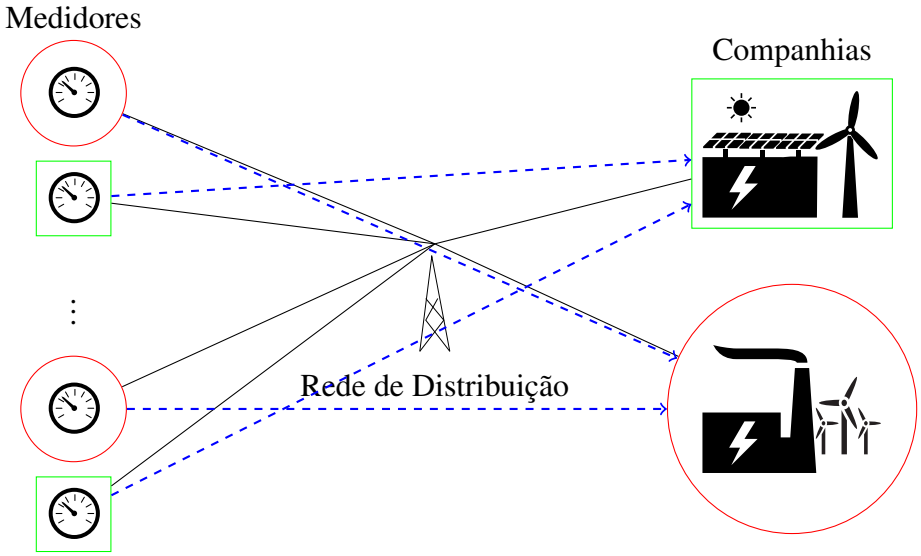

Figura 1.3: Rede elétrica e rede de dados em uma rede inteligente.

1.2.10. Leis sobre Privacidade

Como já vimos os diversos problemas referentes a privacidade, esta seção apresenta resumidamente de que forma as leis estão tratando a privacidade em diferentes culturas. Como já dito, a privacidade é um dos direitos humanos.

> *Ninguém deverá ser submetido a interferências arbitrárias na sua vida privada, família, domicílio ou correspondência, nem ataques à sua honra e reputação. Contra tais intromissões ou ataques todas as pessoas têm o direito à proteção da lei.*[3] [Artigo 12 da Declaração Universal dos Direitos Humanos (1948) [4]]

A legislação brasileira apresenta uma lei específica para privacidade na Internet.

[3]http://www.humanrights.com/pt/what-are-human-rights/videos/right-to-privacy.html
[4]http://www.un.org/en/universal-declaration-human-rights/

> *A garantia do direito à privacidade e à liberdade de expressão nas comunicações é condição para o pleno exercício do direito de acesso à internet.*
> [Art. 8o LEI Nº 12.965, DE 23 DE ABRIL DE 2014.]

O Brasil ainda não apresenta uma consolidação das leis sobre privacidade nos moldes das diretivas de proteção de dados Europeia[5]. No entanto podemos encontrar várias leis referentes a privacidade[6]. Os Estados Unidos da América têm uma regulamentação exacerbada em nível federal e estadual.

Figura 1.4 ilustra como os países são rigorosos quanto a lei de proteção de dados. O mapa é uma cópia de tela[7] gerada em setembro de 2016. No mesmo site, encontra-se um manual com a descrição de cada país.

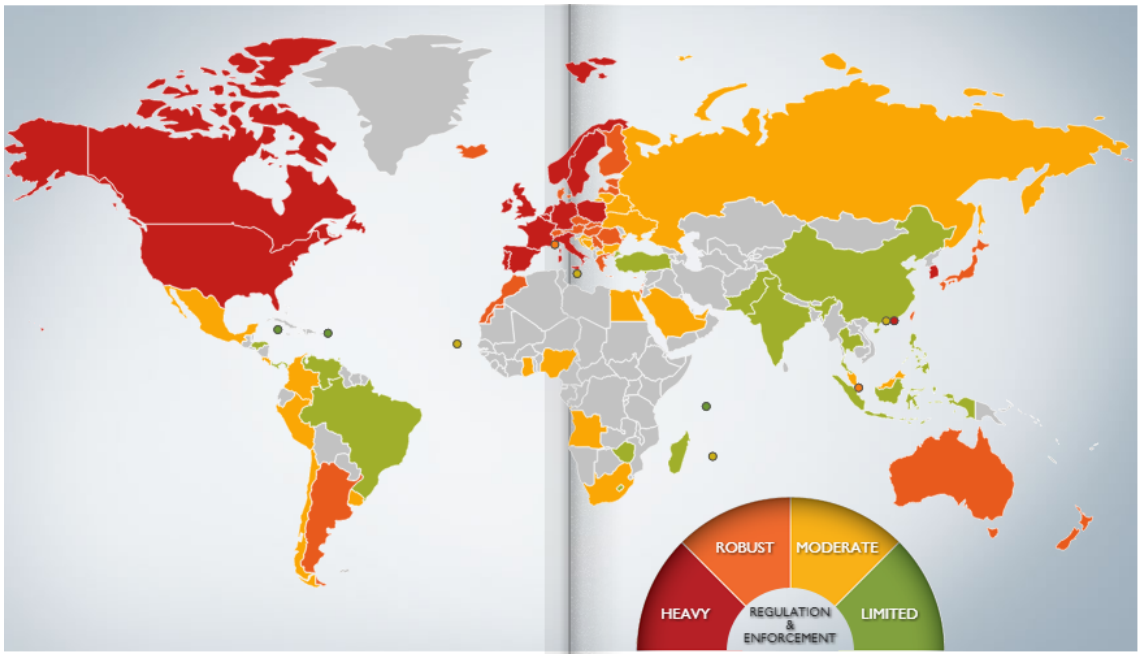

Figura 1.4: Mapa ilustrando como as leis de proteção de dados são tratadas no mundo.

1.3. Métricas para Privacidade

Privacidade diferencial [Dwork 2008], entropia [Díaz et al. 2003] e complexidade algorítmica são métodos usados para determinar se uma técnica criptográfica pode proteger a privacidade. Esta seção apresenta estes métodos e compara-os com uma abordagem introduzida recentemente e que se baseia em probabilidades [Borges de Oliveira 2016]. Assim como outros métodos, privacidade diferencial é usado para validar técnicas de proteção à privacidade, mas não estabelece uma métrica no sentido matemático. No entanto, podemos transformar os métodos para avaliar as técnicas em métricas matemáticas. Esta seção apresenta os métodos usados para avaliar as técnicas para proteger a privacidade. Também se apresenta como transformar estes métodos em métricas. Além disso, esta seção

[5]95/46/EC of 24 October 1995 e 2016/679 of 27 April 2016.
[6]http://dsic.planalto.gov.br/documentos/quadro_legislacao.htm
[7]https://www.dlapiperdataprotection.com/#handbook/world-map-section

apresenta as limitações das técnicas com suas primitivas criptográficas desenvolvidas para proteger a privacidade. Independentemente das técnicas usadas, os problemas de privacidade têm limitações intrinsecamente ligadas a várias variáveis [Borges de Oliveira 2016], por exemplo, se todos os outros participantes de um protocolo que usa uma técnica conspirarem, *i.e.*, conluiarem, os dados de um participante serão revelados independentemente do protocolo ou da técnica.

1.3.1. Anonimização

Além dos métodos já mencionados, temos também k-anonimato, l-diversidade, e t-proximidade [Li et al. 2007]. Todos usam o conceito de um conjunto E de dados indistinguíveis por um identificador em uma tabela. Estes métodos podem ser usados para anonimizar e avaliar a anonimização.

O método de k-anonimato suprime colunas de uma tabela ou troca-as por valores gerais de forma que cada E contenha pelo menos k registros. Este método apresenta sérias limitações visto que 4 pontos determinando a posição no tempo são suficientes para identificar de forma única 95% dos usuários de celular [De Montjoye et al. 2013].

Já l-diversidade requer que cada E tenha pelo menos l valores bem representados para cada coluna sensível. Bem representado pode ser definido de três formas:

1. pelo menos l distintos valores para cada coluna sensível;
2. para cada E, a entropia de Shannon é limitada, tal que $H(E) \geqslant \log_2 l$, onde

$$H(E) = -\sum_{s \in S} \Pr(E,s) \log_2(\Pr(E,s)),$$

   S é o domínio da coluna sensível, e $\Pr(E,s)$ é probabilidade da parte de linhas em E que tem valores sensíveis s;
3. o valor mais comum não pode aparecer com muita frequência e os incomuns não podem aparecer muito raramente.

Note que nem sempre as tabelas têm l distintos valores sensíveis. Além disto, a entropia da tabela tem que ser no mínimo $\log_2 l$. Finalmente, a frequência de valores comuns e incomuns normalmente não tendem a ser próximas.

Um E é dito ter t-proximidade, se a distância entre a distribuição de uma coluna sensível em E e a distribuição da coluna em toda a tabela não é mais do que um valor limiar t. Dizemos que uma tabela tem t-proximidade se todos E na tabela têm t-proximidade. Note que neste caso, t-proximidade gera uma relação inversa entre utilidade dos dados e privacidade.

1.3.2. Modelo de Segurança

Para garantir a privacidade, faz-se necessário garantir a segurança previamente. Certamente, o uso de primitivas criptográficas inseguras não garante a proteção da privacidade. Em particular, se os usuários acreditam que as mensagens estão seguras em um canal de comunicação, eles tendem a enviar mensagens confidenciais. Contrariamente, eles não

enviam mensagens confidenciais se o canal for inseguro. Portanto, a falsa sensação de segurança representa uma ameaça do maior que a transmissão sem proteção.

São diversos os exemplos de falsa sensação de segurança. Um decreto[8] vigente desde 2009 recomenda o uso de um algoritmo de hash chamado MD5. No entanto, o algoritmo já era considerado inseguro desde 2005 [Wang and Yu 2005].

Imagine que uma bolsa de valores ou uma companhia de cartão de crédito envie as transações financeiras por e-mail em um arquivo PDF criptografado com algoritmos seguros. Entretanto, os clientes não podem escolher suas senhas, mas a instituição escolhe as senhas baseadas em sequências numéricas. Esta informação é conhecida por todo cliente. Logo, um adversário poderia saber disto facilmente, e executando o Código 1.1, ele poderia descobrir a senha de um cliente em menos de um minuto. Interceptando os e-mails, o adversário poderia saber todas as transações de todos os clientes enquanto que eles estariam se sentindo seguros porque ninguém consegue quebrar o AES.

Código 1.1 Script em Bash de Força Bruta

```
1: #!/bin/bash
2: #clear
3: echo
4: echo "Processando...␣"
5: for i in {0..999999}
6: do
7:         pdfinfo -upw $i secreto.pdf  &> /dev/null
8:         if [ "$?" -eq "0" ]; then
9:                 echo $i > senha.out
10:                break
11:        fi
12: done
13: cat senha.out
14: xpdf -upw $i secreto.pdf &
```

O modelo de segurança criado por Shannon considera que tanto a geração quanto distribuição das chaves são seguras. O adversário conhece os algoritmos usados na criptografia e tem acesso ao canal de comunicação. Ele apenas não sabe quais são as chaves seguras. Figura 1.5 esquematiza o modelo de segurança criado por Shannon. O adversário tem acesso às mensagens encriptadas $\mathfrak{M}_{i,j}$ pelo usuário i no tempo j, mas não tem acesso às mensagens $m_{i,j}$ decifradas. Ele também conhece a função que encripta Enc e a função que decripta Dec, mas é enfatizado que ele não tem acesso às chaves que devem ser seguras e nem às mensagens $m_{i,j}$.

Note que o modelo foi criado antes do advento da criptografia assimétrica. Porém, é válido para algoritmos de criptografia simétricas e assimétricas.

1.3.3. Modelo de Privacidade

O modelo de segurança não protege a privacidade porque o adversário tem a mesma informação que o destinatário tem. Pode-se, até mesmo considerar que o destinatário é o

[8]http://www.planalto.gov.br/ccivil_03/_ato2007-2010/2009/decreto/D6870.htm

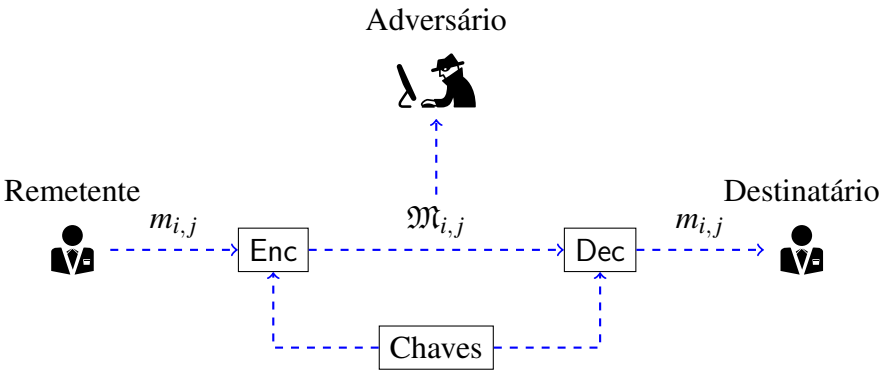

Figura 1.5: Modelo de Shannon para a segurança das mensagens $m_{i,j}$.

adversário. Consequentemente, se o adversário sabe que apenas o usuário i enviou uma mensagem $m_{i,j}$ no tempo j, então a privacidade foi invadida. Mesmo que apenas um usuário i tenha uma mensagem $m_{i,j}$ para ser enviada, faz-se necessário que vários usuários enviem mensagens para que o adversário não identifique o remetente. Se a mensagem $m_{i,j}$ fosse um texto, todos os outros usuários enviariam um texto em branco. Neste caso, o destinatário recebe uma consolidação encriptada $\mathfrak{C}_j$ em vez de receber uma mensagem $m_{i,j}$. Consequentemente, ele pode decriptar a consolidação encriptada $\mathfrak{C}_j$ e ter acesso à consolidação c_j das mensagens no tempo j. Logo, precisamos de um conjunto de usuários i e dizemos que o conjunto tem cardinalidade I, *i.e.*, o número de usuário é número de usuários. Cada consolidação c_j acontece em um tempo j e a cardinalidade do conjunto de consolidações é definida pelo número de tempo J. Certamente, o número de tempo J não tem que ser limitado. Apenas fixamos um número inteiro para podermos definir interessantes momentos em um protocolo. Por exemplo, o tempo j pode ser o ano que ocorreu uma eleição e o número de tempo J pode propiciar uma avaliação das últimas J eleições. De forma mais prática e geral, se a mensagem $m_{i,j}$ é uma transação do usuário i no tempo j, então o número de tempo J pode representar o total de transações em um mês. Normalmente, a consolidação c_j em um tempo j é chamada de agregação e se trata de uma soma. No entanto, elas podem ter significados diferentes. Em redes inteligentes, podemos ter cenários mais complexos e diferenciar a medição agregada da medição consolidada. Neste caso, agregação é uma única medição que engloba vários clientes que moram em uma área, normalmente medida com um *phasor measurement unit* (PMU) em transformadores ou sustações elétricas. Consolidação é a junção de medições individuais dos mesmos clientes um uma única informação, normalmente as medições dos clientes são feitas com medidores inteligentes em suas residências.

Figura 1.6 esquematiza a consolidação de mensagem $m_{i,j}$ de um número de usuários I em um tempo j. O adversário tem todas as informações do destinatário, mas não dos remetentes. Dependendo do protocolo de proteção da privacidade, o adversário pode ou não ter acesso às mensagens encriptadas $\mathfrak{M}_{i,j}$ durante o processo de consolidação. O adversário tem acesso à consolidação encriptada $\mathfrak{C}_j$, mas tal conhecimento não é tão interessante, pois o adversário tem acesso à consolidação c_j, que não tem criptografia.

Note que a ideia central por trás dos métodos de anonimização é a consolidação. Muitas aplicações têm usado alguma primitiva de criptografia homomórfica aditiva

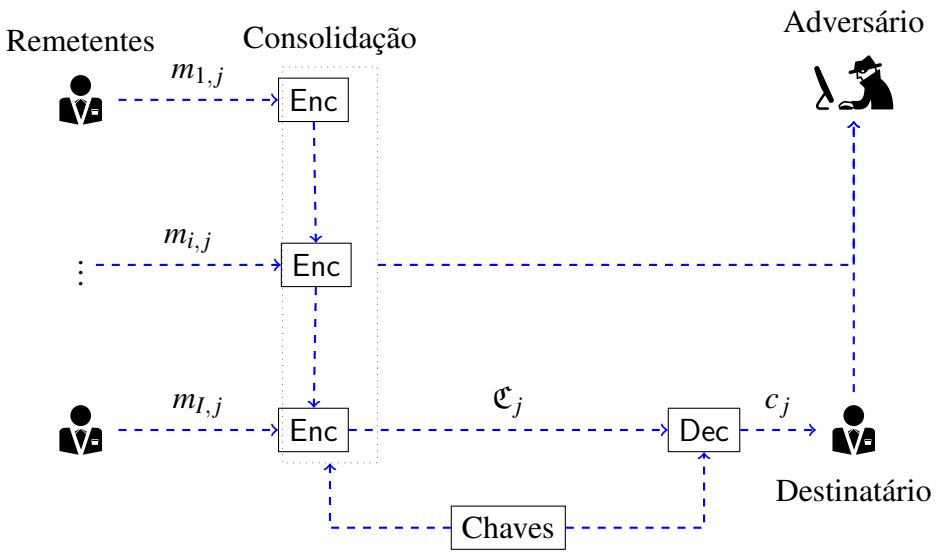

Figura 1.6: Modelo de consolidação das mensagens para preservar a privacidade.

(PCHA). Dizemos que a técnica de criptográfica é PCHA se ela permite descriptografar uma única mensagem $m_{i,j}$ e permite somar mensagens, *i.e.*,

$$\prod_{i=1}^{I} \mathsf{Enc}\left(m_{i,j}\right) = \mathsf{Enc}\left(\sum_{i=1}^{I} m_{i,j}\right) = \mathsf{Enc}\left(c_j\right) = \mathfrak{C}_j.$$

Tais funções devem ser probabilísticas, *i.e.*, mensagens iguais devem resultar mensagens encriptadas diferentes. Se não, o adversário pode identificar mensagens em branco pela sua frequência. Portanto, ele poderia identificar quem mandou ou não uma mensagem.

1.3.4. Definição de Métrica

Métrica é uma função que mede a distância entre dois pontos. Existem vários tipos de métricas. Elas podem nos ajudar a comparar algoritmos e medir a segurança e privacidade.

Primeiramente, vamos nos lembrar como a métrica é definida. Dado um conjunto $\mathscr{C}$ e uma função $d : \mathscr{C} \times \mathscr{C} \to \mathbb{R}^+$, onde $\mathbb{R}^+$ representa o conjunto dos números reais não-negativos, dizemos que d é uma métrica se as seguintes condições são satisfeitas para todo $x, y, z \in \mathscr{C}$:

1. $d(x,y) \geqslant 0$ positivamente definida
2. $d(x,y) = 0 \Leftrightarrow x = y$ identidade
3. $d(x,y) = d(y,x)$ simetria
4. $d(x,z) \leq d(x,y) + d(y,z)$ desigualdade triangular

Métricas podem ser aplicadas a base de dados, sequências de dados, redes com nós e probabilidades associadas aos nós, séries temporais, *etc.* Os diferentes métodos para medir a privacidade foram construídos para cenários diferentes. É possível transferir os

métodos em diversos cenários. Certamente, todas as informações necessárias ao método têm que estar disponíveis. Na sequência, a nomenclatura das aplicações iniciais usadas em cada método é mantida.

1.3.5. Privacidade Diferencial

A noção de privacidade diferencial é semelhante a noção de indistinguibilidade em criptografia. Para defini-la, precisamos de um número real positivo ε e um algoritmo probabilístico $\mathscr{A}$ que pega uma base de dados como entrada. O algoritmo $\mathscr{A}$ é privado ε-diferencialmente se para toda base de dados D_1 e D_2 que difere em um único elemento, e para todos os subconjuntos de S da imagem de $\mathscr{A}$, temos

$$\Pr[\mathscr{A}(D_1) \in S] \leq e^{\varepsilon} \times \Pr[\mathscr{A}(D_2) \in S],$$

onde a probabilidade é controlada pela aleatoriedade usada pelo algoritmo.

Note que privacidade diferencial não é uma métrica no sentido matemático. Porém, se os algoritmos manterem a probabilidade conforme a entrada, podemos construir uma métrica para comparar a distância entre dois algoritmos calculando

$$d(\mathscr{A}_1, \mathscr{A}_2) = |\varepsilon_1 - \varepsilon_2|.$$

Assim, podemos determinar se dois algoritmos são equivalentes $\varepsilon_1 = \varepsilon_2$, ou ainda, podemos determinar a distância a um algoritmo ideal

$$d(\mathscr{A}_1, \mathscr{A}_{\text{ideal}}) = |\varepsilon_1 - 0|.$$

1.3.6. Entropia

O grau de anonimato g pode ser medido com a entropia de Shannon

$$H(X) = \sum_{i=1}^{N} \left[p_i \cdot \log_2 \left(\frac{1}{p_i} \right) \right],$$

onde $H(X)$ é a entropia da rede, N é o número de nós e p_i é a probabilidade associada a cada nó i. A entropia máxima ocorre quando temos uma probabilidade uniforme, *i.e.*, todos os nós sãs equiprováveis $1/N$, o que gera

$$H_M = \log_2(N).$$

Logo, o grau de anonimato g é definido por

$$g = 1 - \frac{H_M - H(X)}{H_M} = \frac{H(X)}{H_M}.$$

Similarmente, podemos construir uma métrica para comparar a distância entre duas redes calculando

$$d(g_1, g_2) = |g_1 - g_2|.$$

Analogamente, podemos determinar se elas são equivalentes $g_1 = g_2$. Consequentemente, podemos determinar a distância a uma rede de anonimato ideal

$$d(g_1, g_{\text{ideal}}) = |g_1 - 1|.$$

A rede pode ser trocada por um banco de dados, mas neste modelo, cada registro tem que ter uma probabilidade associada a ele.

1.3.7. Complexidade

Análise de complexidade também pode se transformar em uma métrica para medir o tempo necessário para quebrar um algoritmo criptográfico que protege a privacidade. A medida pode ser de forma assintótica ou em número de passos.

De forma geral, mesmo que a complexidade impeça a quebra do algoritmo devido a um problema matemático, mesmo que o algoritmo nos forneça uma privacidade diferencial ideal, e mesmo que o grau de anonimato seja máximo, em todas estas situações a privacidade pode ser violada. Por exemplo, considere uma votação com 4 eleitores, se 3 conspirarem, a privacidade do quarto será violada independentemente de qualquer algoritmo ou protocolo. [Borges de Oliveira 2017e] apresenta como quebrar protocolos baseados em ruído que tem privacidade diferencial para redes inteligentes.

O algoritmo criptográfico deve garantir a privacidade da mesma forma que garante segurança. Uma mensagem cifrada deve ter medidas de confidencialidade máximas para a privacidade, assim como tem para a criptografia. Desta forma, devemos usar a complexidade do melhor algoritmo que resolve um problema matemático associado ao algoritmo criptográfico. Logo, a complexidade pode ser usada para determinar o atual nível de segurança e privacidade de uma mensagem cifrada. Além disto, devemos considerar os ataques a privacidade que são independentes do algoritmo criptográfico.

1.3.8. Contagem e Probabilidades

Esta seção aborda casos de ataques independentes dos algoritmos escolhidos. Logo, temos que contar quantas opções seriam possíveis e quais suas probabilidades.

Voltando ao exemplo da votação, suponha que os 4 eleitores votaram em 3 sins e 1 não. Neste caso, um adversário sabe que a probabilidade de um eleitor ter votado sim é $3/4$ e de $1/4$ para não. Se tivéssemos mais eleitores e o resultado fosse 15 sins e 5 nãos, então as respectivas probabilidades também seriam $3/4$ e $1/4$. A eleição é mais fácil de contar porque o voto é binário, ou eu voto ou não em um candidato. A mesma lógica se aplica ao caso de vários candidatos.

Diferente dos casos binários, pode-se desejar manter a privacidade de valores oriundos de medidas. Para um adversário descobrir uma série temporal de três pontos, ele pode representar cada ponto por um número de estrelas, *i.e.*, o total de símbolos $\star$. Assim, o adversário pode separar o total de estrelas em três caixas. Suponha que a soma da série seja 7, então uma possibilidade seria $\boxed{\star\star\star\star}\,\boxed{\star}\,\boxed{\star\star}$. Por simplicidade, o adversário pode separar as estrelas por barras em vez de caixas. Logo, $\star\star\star\star\ |\star|\star\star$ tem a mesma solução. Com esta notação, a combinação de 7 estrelas mais 2 barras escolhidas 7 estrelas determina o número possível de soluções, matematicamente temos

$$\binom{7+2}{7} = \frac{9!}{7!(9-7)!} = 36.$$

De forma geral, se t é o número total de pontos da série temporal e se s sua soma, então o número de possíveis séries temporais para o adversário decidir a correta é

determinado por s mais $t-1$ escolhendo s, portanto

$$\binom{s+t-1}{s} = \frac{(s+t-1)!}{(t-1)!s!} = \binom{s+t-1}{i-1}. \quad (1)$$

Múltiplas séries temporais podem formar uma tabela, *e.g.*, a lista de candidatos com votos por estados. Para evitar que o candidato eleito privilegie estados, o tribunal eleitoral poderia divulgar apenas o número de votantes por estado e o total de votos por candidato. No entanto, os candidatos poderiam inferir os possíveis votos por estado [Borges de Oliveira 2017e] e dados de eleições passadas poderiam ajudar. Note que tais somas poderiam ser computadas com dados criptografados de forma muito mais robusta que anonimização por k-anonimato, l-diversidade e t-proximidade. Mesmo assim, dependendo do tamanho da tabela e de seus valores, os valores podem ser encontrados.

Em geral, podemos abstrair os votos e outros valores para uma medição. Desta forma, técnicas de anonimização tentam reduzir o número de medições em tabelas. Contrariamente a intuição, quanto menor o número de medições, maior a chance de descobrilas [Borges de Oliveira 2017e].

1.4. Técnicas Simétricas e outras Tecnologias

Além de diversas técnicas que usam criptografia, diversas tecnologias têm sido desenvolvidas para proteger a privacidade, por exemplo, o uso de baterias e outros buffers em smart grids. Esta seção apresenta técnicas baseadas em ruído, pseudônimo e redes de anonimato, e DC-Net simétrica (SDC-Net) introduzidas por [Chaum 1988] cujo nome vem de *Dining Cryptographers Network* (DC-Net). Antes de cada técnica, apresenta-se o fundamento matemático necessário para entender a técnica. Apresenta-se também as limitações das técnicas e tecnologias. Um problema das técnicas simétricas é o número de chaves necessárias para troca de mensagens. Especificamente, o número de chaves cresce quadraticamente com o número de participantes. Existem tecnologias viáveis e inviáveis. O uso de caixas de água em residências para proteger a privacidade em uma smart grid que controla o fluxo de água é viável. Já o uso de baterias para proteger a privacidade em uma smart grid que controla o fluxo da energia elétrica é financeiramente inviável. Técnicas que usam criptografia são muito mais acessíveis. Em particular, SDC-Net podem garantir uma segurança incondicional, mas neste caso, não são adequadas para aplicações reais devido a limitação na qual a chave criptográfica somente pode ser usada uma vez.

1.4.1. Ruído

Várias técnicas de anonimização usam uma função pseudoaleatória ou uma distribuição gaussiana ou laplaciana para introduzir um ruído nas mensagens dos usuários. Em geral, sabe-se para onde a soma dos ruídos convergem. [Borges de Oliveira 2017h] mostrou que tais técnicas não funcionam para smart grids. Tais técnicas não funcionam porque um adversário poderia usar várias medidas com ruído de um determinado horário e a soma das medidas com ruído convergem para soma sem ruído. Logo, um adversário poderia criar um perfil do cliente.

Em geral, um adversário pode descobrir uma mensagem encriptada $\mathfrak{M}_{i,j}$ para qualquer aplicação de ruído onde se saiba sua convergência. Suponha que $\mathfrak{M}_{i,j} = m_{i,j} +$

$r_{i,j}$ é a mensagem $m_{i,j}$ com ruído $r_{i,j}$ de usuário i no tempo j. Por simplicidade, suponha que a distribuição tenha valor esperado $\mu = 0$, logo a série de ruídos converge para zero

$$\sum_{i}^{\infty} r_{i,j} = \sum_{j}^{\infty} r_{i,j} \to 0.$$

De fato, a série aproxima de zero caso ela seja truncada com um número suficiente de termos. Desta forma, um adversário gera uma série de ruídos r'_l com a mesma distribuição e computa

$$\mathfrak{M}_{i,j} + \sum_{l=1}^{L} r'_l \to m_{i,j},$$

onde L é um número suficientemente grande.

1.4.2. Pseudônimo e Redes de Anonimato

Em vez de usar a verdadeira identidade, usar um pseudônimo para cada serviço é uma forma de manter anonimato. No entanto, um adversário pode rastrear o meio de comunicação e identificar um usuário pela origem da conexão. Por isto, faz-se necessário que a conexão venha de uma rede de anonimato, *i.e.*, um adversário não consegue rastrear a origem da conexão.

Redes de anonimato podem ser construídas de forma física, *e.g.*, todas as máquinas conectadas no mesmo barramento recebendo todas as mensagens. Assim, o adversário apenas sabe que o pseudônimo é usado em uma daquelas máquinas. Além disto, redes de anonimato podem ser construídas por software. [Chaum 1981] introduziu o conceito de Mix networks que são redes de anonimato construídas por servidores proxy chamados de mixes.

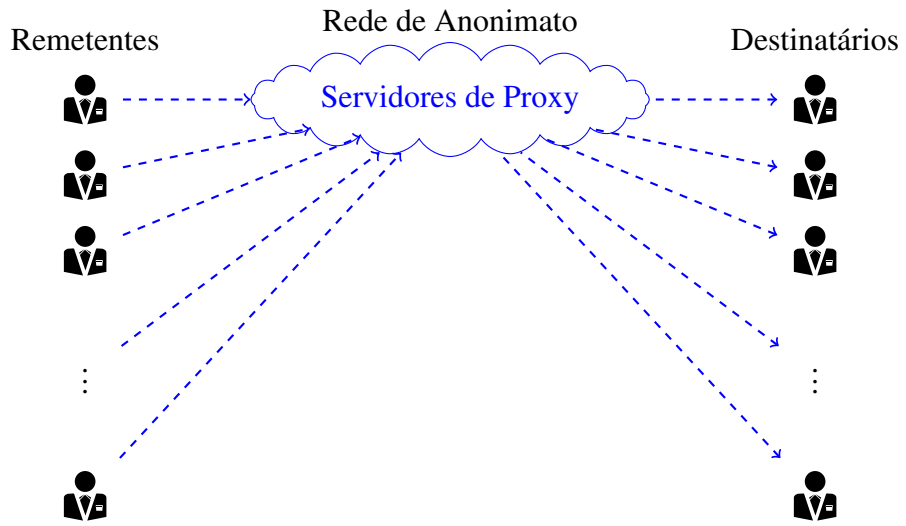

Figura 1.7: Uma mix network com possíveis remetentes e possíveis destinatários.

[Borges et al. 2012] apresentam uma relação inversa entre performance e privacidade em redes de anonimato que usam pseudônimos. Desta forma, funções criptográficas homomórficas são mais eficientes que usar redes de anonimato e pseudônimos. Tais

funções permitem manipular mensagens criptografadas, *i.e.*, podemos aplicar operações matemáticas e fazer buscas em dados cifrados. Redes de anonimato podem ser construídas sem a necessidade de terceiros como servidores de proxy. SDC-Net podem ser usadas como redes de anonimato.

1.4.3. DC-Nets Simétricas

[Chaum 1988] introduziu o conceito de SDC-Net através de um problema de privacidade com sua respectiva solução. Resumidamente, três criptógrafos foram pagar a conta do jantar, mas a conta já estava paga. Eles queriam saber se alguma agência de segurança pagou ou se um deles pagou, mas sem revelar a identidade de quem pagou. Primeiramente eles combinam uma senha com os outros dois. Figura 1.8 esquematiza a distribuição de chaves para SDC-Net.

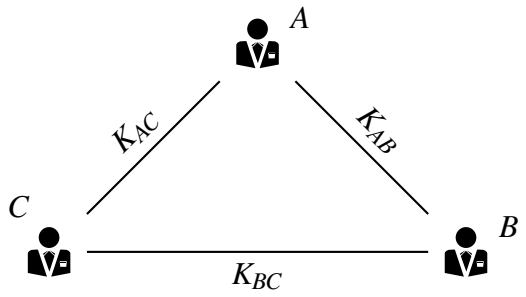

Figura 1.8: Distribuição de chaves em SDC-Net.

Como a porta lógica XOR $\oplus$ cancela sequências igual de bits, *e.g.*, $101101 \oplus 101101 = 000000$. Assim, $k \oplus 0 = k$ e $k \oplus 1 \oplus k = 1$ para todo k. Além disto, $K_{AB} \oplus K_{AC}$ é conhecido apenas pelo A. Sem perda de generalidade, suponha que A pagou. Logo, A revela $K_{AB} \oplus K_{AC} \oplus 1$, enquanto B revela $K_{AB} \oplus K_{BC}$, e C revela $K_{AC} \oplus K_{BC}$. Todos podem calcular

$$K_{AB} \oplus K_{AC} \oplus 1 \oplus K_{AB} \oplus K_{BC} \oplus K_{AC} \oplus K_{BC} = 1.$$

Caso ninguém tivesse pago a conta seria

$$K_{AB} \oplus K_{AC} \oplus K_{AB} \oplus K_{BC} \oplus K_{AC} \oplus K_{BC} = 0.$$

Sem perda de generalidade, suponha agora que o usuário A do protocolo é um adversário e calcula

$$K_{AB} \oplus \underbrace{(K_{AB} \oplus K_{BC})}_{\text{revelado por } B} = K_{BC}$$

e

$$K_{Ac} \oplus \underbrace{(K_{AC} \oplus K_{BC})}_{\text{revelado por } C} = K_{BC}.$$

Portanto, a chave não pode ser usada novamente.

Se A não tivesse enviado a mensagem que pagou, ou A teria obtido

$$K_{AB} \oplus \underbrace{(K_{AB} \oplus K_{BC} \oplus 1)}_{\text{revelado por } B} = K_{BC} \oplus 1$$

e

$$K_{Ac} \oplus \underbrace{(K_{AC} \oplus K_{BC})}_{\text{revelado por } C} = K_{BC},$$

ou A teria obtido

$$K_{AB} \oplus \underbrace{(K_{AB} \oplus K_{BC})}_{\text{revelado por } B} = K_{BC}$$

e

$$K_{Ac} \oplus \underbrace{(K_{AC} \oplus K_{BC} \oplus 1)}_{\text{revelado por } C} = K_{BC} \oplus 1.$$

Note que A não tem uma informação sobre quem pagou, que já não tinha anteriormente. Se as senhas forem verdadeiramente aleatória e forem usadas apenas uma vez, então temos um segredo perfeito para privacidade.

Para evitar que a chave K_{BC} acordada entre B e C seja revelada, eles podem usar uma função de hash H com um tempo j para proteger as chaves em vez deles usarem diretamente as chaves. A chave pode ser concatenada junto com o tempo j, e então, calculada a função de hash H, *e.g.*, $\mathsf{H}(K_{AB}||j)$ em vez de apenas K_{AB}, o símbolo $||$ significa concatenação. Desta forma, o mesmo ataque revelaria apenas $\mathsf{H}(K_{BC}||j)$. Como o tempo j não se repete, a função de hash geraria a aleatoriedade necessária para proteger a chave.

Em vez de usarmos uma senha para cada par de usuários, cada usuário pode mandar uma senha para cada um dos outros. No exemplo prévio, as conexões entre os usuários formam um grafo, mas agora forma um grafo orientado. Não necessariamente, todos os usuários teriam que estabelecer chaves com todos os outros. Usuários poderiam ter chaves apenas com os usuários de sua confiança. Figura 1.9 esquematiza uma SDC-Net formando um dígrafo completo, *i.e.*, orientado.

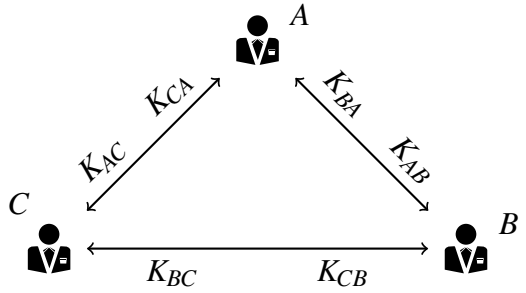

Figura 1.9: Distribuição de chaves em SDC-Net formando um dígrafo completo.

Similarmente, se A for um adversário, então A descobriria $\mathsf{H}(K_{BC}||j) \oplus \mathsf{H}(K_{CB}||j)$. Note que geraria mais aleatoriedade com duas funções de hash.

Se um dos usuários um adversário, ele pode apenas inverter sua mensagem, *i.e.*, calcular XOR 1 com sua mensagem, gerando no final das contas $1 \oplus 1 = 0$. Assim, os outros pensariam que um deles pagou quando nenhum deles pagou, e vice-versa. Somente quem pagou poderia perceber que há algo errado, mas revelando que há algo errado, revelaria sua identidade.

Poderíamos ter um problema diferente. Imagine que os três estão no elevador que somente permite levar 250Kg. Eles precisam saber a soma de seus pesos, mas não querem revelar seus respectivos pesos. Poderíamos querer saber a média da idade das pessoas que trabalham como criptógrafos, mas sem revelar idades individuais. Nestes casos, poderíamos usar uma operação de soma em vez de usarmos um XOR. Bastaria que os criptógrafos somassem as chaves enviadas e subtraíssem as chaves recebidas, ou vice-versa. [Borges de Oliveira 2017h] apresenta como SDC-Net pode ser usada em redes inteligentes.

Podemos generalizar como SDC-Nets funcionam. Inicialmente, cada usuário i envia uma chave para cada um dos outros. Para cada tempo j, cada usuário calcula

$$\mathfrak{M}_{i,j} = m_{i,j} + \sum_{o \in \mathscr{U} - \{i\}} \mathsf{H}\left(k_{i,o} || j\right) - \mathsf{H}\left(k_{o,i} || j\right),$$

onde $\mathscr{U}$ é o conjunto de usuários, $k_{i,o}$ é a chave enviada por i a o, e $k_{o,i}$ é a chave enviada por o a i.

Um usuário poderia ser o responsável por consolidar, descriptografar e revelar a consolidação c_j. No entanto, todos podem fazer isto executando uma SDC-Net. Além disto, o processo de consolidação já decripta, *i.e.*,

$$c_j = \sum_{i=1}^{I} \mathfrak{M}_{i,j}.$$

Independente da aplicação, SDC-Net permite que um adversário cause um rompimento, *i.e.*, um adversário pode inserir um valor falso invalidando o resultado do protocolo. Outro problema é o número de chaves que cresce quadraticamente, *i.e.*, para o caso da rede orientada temos $I(I-1)$ chaves, onde I é número de usuários, e para o caso da rede não-orientada temos

$$\frac{I(I-1)}{2}$$

chaves. Similarmente, o número total de operações que os usuários têm que calcular cresce quadraticamente com o número de usuários I. Para o caso da rede orientada temos $2I(I-1)$ operações e para o caso da rede não-orientada temos $I(I-1)$ operações, sem contarmos as operações referentes as mensagens.

Para diminuir a complexidade na distribuição de chaves e no processamento, os usuários podem enviar chaves apenas para quem eles confiam. Figura 1.10 esquematiza uma distribuição de chaves baseada em confiança. Note que o usuário B pode decifrar tudo que o usuário E encripta. Mas, ninguém pode decifrar as mensagens do usuário E sem a senha do usuário B. Os usuários têm que calcular menos funções de hash. O mais leve é E que calcula apenas uma função de hash, D calcula duas, C quatro, A cinco, e B seis. Basear a segurança em confiança não é uma boa escolha para um protocolo, mas é uma opção. Conectividade máxima evita conluio, mas tem um custo computacional.

Apesar da complexidade e da possibilidade de rompimento do protocolo, SDC-Net tem uma grande vantagem. Enquanto que PCHA permite que o vazamento de apenas uma mensagem encriptada $\mathfrak{M}_{i,j}$ para revelar a mensagem $m_{i,j}$ e o valor consolidado pode

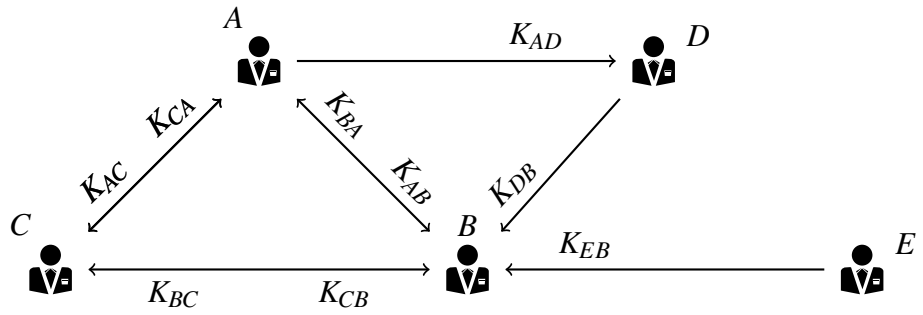

Figura 1.10: Distribuição de chaves em SDC-Net orientada.

não conter todas as contribuições, SDC-Net apresenta um cenário melhor. No caso do grafo completo, SDC-Net é resistente a conluio. Todos menos um $I-1$ devem conspirarem para descobrirem as mensagens de um usuário. Outra grande vantagem é que existe garantias que a informação consolidada somente será revelada com a contribuição de todos os usuários. SDC-Net apresenta o melhor cenário para a privacidade.

1.5. Técnicas Assimétricas e de Compromisso

Esta seção apresenta três técnicas: compromisso (*commitment*), encriptação homomórfica e DC-Nets assimétricas (ADC-Nets). O conhecimento matemático necessário para entender estas técnicas reside em estruturas algébricas que serão apresentadas antes das técnicas. Assim como funções de hash, técnicas de compromisso não são classificadas como simétricas ou assimétricas. Estão nesta seção apenas pela proximidade das ferramentas matemáticas usadas em técnicas de commitment e criptografia assimétrica. Em particular, o número de chaves de técnicas assimétricas cresce linearmente com o número de participantes. Entretanto, técnicas de commitment não tem chave criptográfica, mas precisam apenas armazenar um valor que se equipara a uma chave. Este valor é usado para provar—no sentido de garantir ou verificar—a mensagem. Diferente de técnicas criptográficas que se cifra uma mensagem e depois a decifra, commitment garante que o emissor de uma mensagem não vai alterar seu conteúdo, mas o receptor não o conhece até o emissor apresentá-lo. Logo, o receptor pode verificar uma soma sem conhecer as parcelas somadas com uma técnica de commitment homomórfico, por exemplo. Após a apresentação de técnicas de commitment, detalha-se Pedersen Commitments [Pedersen 1992] que garante uma segurança incondicional em relação a um commitment, ou seja, uma mensagem comprometida por um código criptográfico. Diferente da segurança incondicional em SDC-Nets, Pedersen Commitments é viável. Similarmente à apresentação de commitment, após a apresentação de técnicas de encriptação homomórfica, detalha-se o esquema de Paillier [Paillier 1999] que é uma técnica de encriptação homomórfica aditiva, e por isto, ela é usada em vários cenários. A seção termina detalhando SDC-Nets [Borges de Oliveira 2016]. Em particular, as equações de ADC-Nets podem ser derivadas das equações de técnicas de encriptação homomórfica aditiva. Logo, há semelhanças nas equações das técnicas, entretanto as propriedades das técnicas diferem em vários aspectos. O mesmo acontece com Pedersen Commitments que tem equação similar, mas não pode nem ser considerado um esquema que encripta e decripta.

1.5.1. Compromisso (Commitment)

Diferente de um sistema criptográfico que encripta e decripta mensagens $m_{i,j}$, técnicas de compromisso usam uma função de comprometimento Commit e uma função de abertura Open junto com um verificador randômico $\mathfrak{v}_{i,j}$, *i.e.*,

$$\mathfrak{N}_{i,j} = \mathsf{Commit}(m_{i,j}, \mathfrak{v}_{i,j})$$

e

$$\mathsf{Open}(\mathfrak{N}_{i,j}, m_{i,j}, \mathfrak{v}_{i,j}) = \top \veebar \bot,$$

onde ou a função de abertura retorna verdadeiro $\top$ ou retorna falso $\bot$. Desta forma, não é possível descriptografar a mensagem $m_{i,j}$, mas se pode verificar quando a mensagem $m_{i,j}$ é correta ou não.

Um exemplo de aplicação, seria um testamento $m_{i,j}$. Dado $\mathfrak{N}_{i,j}$ não se saberia qual é o testamento $m_{i,j}$, mas poderia verificar se ele é verdadeiro ou não. Sempre que pensarmos em auditoria, verificação, averiguação, podemos achar uma aplicação para técnicas de compromisso.

Em transações financeiras estamos interessados em verificar um total, *e.g.*, queremos verificar se a conta fecha no fim do mês. Neste caso, podemos proteger a privacidade dos usuários revelando apenas o valor final, *i.e.*, sem revelar as mensagens $m_{i,j}$ individuais. Abstratamente, cada usuário i no tempo j aplica a função

$$\mathfrak{N}_{i,j} = \mathsf{Commit}(m_{i,j}, \mathfrak{v}_{i,j})$$

e manda $\mathfrak{N}_{i,j}$ o resultado para um responsável pela verificação, digamos um auditor. Ao final de um período de tempo, *e.g.*, um mês, o auditor calcula

$$\mathfrak{U}_i = \prod_{j=1}^{J} \mathfrak{N}_{i,j}$$

enquanto cada usuário calcula

$$b_i = \sum_{j=1}^{J} m_{i,j}$$

e

$$\mathfrak{V}_i = \sum_{j=1}^{J} \mathfrak{v}_{i,j}$$

e envia $\mathfrak{U}_i$ e $\mathfrak{V}_i$ para o auditor, onde b_i é o valor da conta e $\mathfrak{U}_i$ é um verificador do usuário i gerado randomicamente. Para que o auditor verifique que b_i é o valor correto e que o usuário i mostre que é o correto, basta que eles calculem

$$\mathsf{Open}\left(\mathfrak{U}_i, b_i, \mathfrak{V}_i\right)$$

e verifiquem se o resultado é verdadeiro $\top$ ou se é falso $\bot$.

Se o auditor quisesse verificar a consolidação c_j para o número de usuários I, bastaria calcular

$$\mathfrak{T}_j = \prod_{i=1}^{I} \mathfrak{N}_{i,j}$$

e pedir o verificador no tempo j

$$\mathfrak{R}_j = \sum_{i=1}^{I} \mathfrak{v}_{i,j}.$$

Os usuários poderiam calcular o verificar com uma SDC-Net. Com as informações, o auditor verifica se

$$\mathsf{Open}\left(\mathfrak{T}_j, c_j, \mathfrak{R}_j\right)$$

retorna verdadeiro $\top$ ou se retorna falso $\bot$.

Em geral, um algoritmo de comprometimento não deve ser executado sozinho, mas junto com uma função de assinatura Sign que forneça uma assinatura digital $\mathfrak{S}_{i,j}$. Figura 1.11 esquematiza o modelo de comunicação entre usuários e auditor, que poderia ser um dos usuários. Cada mensagem comprometida

$$\mathfrak{N}_{i,j} = \mathsf{Commit}(m_{i,j}, \mathfrak{v}_{i,j})$$

deve ir concatenada a sua respectiva assinatura digital

$$\mathfrak{S}_{i,j} = \mathsf{Sign}\left(\mathfrak{N}_{i,j}\right).$$

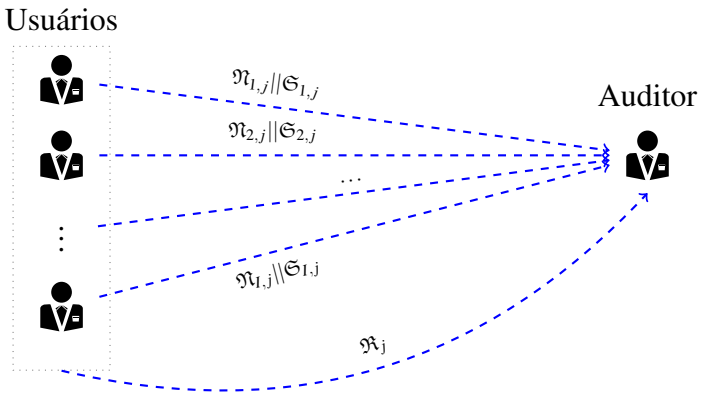

Figura 1.11: Modelo de comunicação para verificação no tempo j.

Entres as técnicas de comprometimento, existe uma que se sobressai porque pode prover segurança incondicional [Pedersen 1992]. Mas, antes precisamos saber que um grupo é um conjunto com uma operação que tem associatividade, identidade e inversa. Denotamos por $\mathbb{Z}_p$ o grupo multiplicativo dos números inteiros módulo um número primo p.

Antes dos usuários rodarem a função de comprometimento Commit, eles precisam estipular alguns valores em uma fase de inicialização, onde eles escolhem dois primos p e q tal que $q|(p-1)$ e sejam suficientemente grandes, e dado um gerador g de ordem q do subgrupo $G \subset \mathbb{Z}_p^*$, e deixam os valores públicos. Na sequência, cada usuário escolhe secretamente um $a_i \in \mathbb{Z}_q$ e calcula

$$h_i = g^{a_i} \pmod{p}$$

depois envia h_1 para o auditor. Note que dado h_i, g, e p, não se sabe da existência de um algoritmo clássico com tempo polinomial que determine a_i, *i.e.*, determinar a_i é um problema intratável para computadores clássicos. Porém, isso pode ser resolvido com o advento dos computadores quânticos [Shor 1997].

Para enviar a mensagem comprometida $\mathfrak{N}_{i,j}$, os usuários escolhem aleatoriamente outro valor secreto $\mathfrak{v}_{i,j} \in \mathbb{Z}_q$ e calculam

$$\mathfrak{N}_{i,j} = \mathsf{Commit}\left(m_{i,j}, \mathfrak{v}_{i,j}\right) = g^{m_{i,j}} h_i^{\mathfrak{v}_{i,j}} \mod p,$$

onde $m_{i,j} \in \mathbb{Z}_q$.

Para abrir, o auditor precisa da mensagem $m_{i,j}$ e do verificador randômico $\mathfrak{v}_{i,j}$. A função de abertura é definida por

$$\mathsf{Open}\left(\mathfrak{N}_{i,j}, m_{i,j}, \mathfrak{v}_{i,j}\right) = \left(\mathfrak{N}_{i,j} \stackrel{?}{=} g^{m_{i,j}} h_i^{\mathfrak{v}_{i,j}} \mod p\right).$$

Para garantir a privacidade a verificação deve acontecer em uma consolidação. Por exemplo, verificar o total das mensagens do usuário. Logo, o auditor calcula

$$\mathfrak{U}_i = \prod_{j=1}^{J} \mathfrak{N}_{i,j} = \prod_{j=1}^{J} g^{m_{i,j}} h_i^{\mathfrak{v}_{i,j}} \mod p$$

enquanto o usuário i calcula

$$b_i = \sum_{j=1}^{J} m_{i,j}$$

e

$$\mathfrak{V}_i = \sum_{j=1}^{J} \mathfrak{v}_{i,j}.$$

Na sequência, o usuário i envia $\mathfrak{U}_i$ e $\mathfrak{V}_i$ para o auditor. Para verificar se as somas estão corretas, basta que eles calculem

$$\mathsf{Open}\left(\mathfrak{U}_i, b_i, \mathfrak{V}_i\right) = \left(\mathfrak{U}_i \stackrel{?}{=} g^{b_i} h_i^{\mathfrak{V}_i} \mod p\right)$$

e verifiquem se o resultado é verdadeiro $\top$ ou se é falso $\bot$.

Para o auditor verificar a consolidação c_j no número de usuários I, temos um problema. O valor de h_i teria que ser igual para todos os usuários i. Mas, a_i deveria ser secreto. Então voltando a fase inicial, cada usuário poderia escolher seu a_i e usando uma SDC-Net, eles poderiam revelar o produto h s.t.

$$h = \prod_{i=1}^{I} g^{a_i}.$$

Então, eles geram um único h sem revelarem seus exponentes secretos a_i.

Desta forma, o auditor verifica a consolidação c_j no tempo j calculando

$$\mathfrak{T}_j = \prod_{i=1}^{I} \mathfrak{N}_{i,j} = \prod_{i=1}^{I} g^{m_{i,j}} h^{\mathfrak{v}_{i,j}} \mod p,$$

enquanto os usuários usam uma SDC-Net para calcular verificador no tempo j

$$\mathfrak{R}_j = \sum_{i=1}^{I} \mathfrak{v}_{i,j}.$$

Finalmente, eles verificam

$$\mathsf{Open}\left(\mathfrak{T}_j, c_j, \mathfrak{R}_j\right) = \left(\mathfrak{T}_j \stackrel{?}{=} g^{c_j} h^{\mathfrak{R}_j}\right)$$

retorna verdadeiro $\top$ ou se retorna falso $\bot$. A multi-exponenciação modular $g^{m_{i,j}} h^{\mathfrak{v}_{i,j}} \bmod p$ pode ser calculada rapidamente e paralelizada com ótimo balanceamento de cargas [Borges et al. 2017]. Para calcular a exponenciação ou multi-exponenciação modular, pode-se usar o Algoritmo 1 do apêndice A.

O processo de verificação por esquemas de compromissos pode ser feito de diversas maneiras. Um subconjunto de usuários poderia ser verificado. As consolidações no tempo poderiam ser semanais, mensais e anuais. Apenas, deve-se tomar cuidado que um excessivo número de verificações não leve a vazamentos de privacidade.

Um aspecto interessante de implementação é a geração de $\mathfrak{U}_i$, $\mathfrak{T}_j$, $\mathfrak{V}_i$, $\mathfrak{R}_j$, b_i e c_j pois o software não precisa armazenar todos os valores para gerá-los. É necessário armazenar apenas um valor para cada um formando um acumulador, *e.g.*, fazemos $c_j = 0$ e depois $c_j \leftarrow c_j + m_{i,j}$. Portanto, não se faz necessário armazenar todas as mensagens $m_{i,j}$. Após a verificação, também podemos eliminar c_j.

1.5.2. Encriptação Homomórfica

Existem dois tipos de encriptação homomórfica, a saber, parcialmente e completamente homomórficas. A primeira propicia uma das duas operações ou soma ou multiplicação sobre mensagem encriptada $\mathfrak{M}_{i,j}$. Enquanto que a segunda propicia ambas operações, *i.e.*, soma e multiplicação sobre mensagem encriptada $\mathfrak{M}_{i,j}$. Quando a técnica criptográfica apenas possibilita a operação de soma, dizemos que ela é uma primitiva de criptografia homomórfica aditiva (PCHA).

Atualmente, a maioria dos problemas de privacidade podem ser resolvidos apenas com PCHA. Logo, esta seção está focada em um sistema criptográfico desenvolvido por [Paillier 1999] que nos PCHA. Diferente de técnicas de compromisso que usam apenas um verificador randômico $\mathfrak{v}_{i,j}$, Paillier tem uma chave pública e outra privada. Logo, alguém seria responsável pela chave privada, digamos um contador.

Em uma fase inicial, o contador escolhe dois primos p e q grandes o suficiente de forma aleatória. Na sequência, o contador calcula $n = p \cdot q$ e $\lambda = \mathsf{mmc}(p-1, q-1)$, e escolhe aleatoriamente um número $g \in \mathbb{Z}^*_{n^2}$ s.t. n divide a ordem de g. Desta forma, o contador tem a chave privada (λ, μ) e distribui a chave pública (n, g) para os usuários, que podem encriptar com a função

$$\begin{aligned} &\mathsf{Enc} : \mathbb{Z}_n \times \mathbb{Z}^*_n \to \mathbb{Z}_{n^2} \\ &\mathsf{Enc}(m_{i,j}, r_{i,j}) \mapsto g^{m_{i,j}} \cdot {\mathfrak{v}_{i,j}}^n \mod n^2, \end{aligned} \tag{2}$$

onde $\mathfrak{v}_{i,j}$ é um valor randômico secreto. Normalmente, não escrevemos $\mathfrak{v}_{i,j}$. Logo, denotamos apenas $\mathfrak{M}_{i,j} = \mathsf{Enc}\left(m_{i,j}\right)$.

Para proteger a privacidade com PCHA, algum agregador semi-honesto deve entrar em ação e calcular

$$\mathfrak{C}_j = \sum_{i=1}^{I} \mathfrak{M}_{i,j}.$$

Semi-honesto significa que não forma um conluio e pode querer ler as mensagens $m_{i,j}$, mas o agregador não consegue decifrar as mensagens encriptadas $\mathfrak{M}_{i,j}$.

Para descriptografar o contador aplica a função

$$\begin{aligned} &\mathsf{Dec} : \mathbb{Z}_{n^2} \to \mathbb{Z}_n \\ &\mathsf{Dec}(\mathfrak{C}_j) \mapsto \mathrm{L}({\mathfrak{C}_j}^{\lambda} \mod n^2) \cdot d \mod n, \end{aligned} \tag{3}$$

onde $d = L(g^{\lambda} \mod n^2)^{-1}$.

Para se calcular a inversa multiplicativa em um grupo $\mathbb{Z}_n$, *i.e.*, de um inteiro módulo n, pode-se usar o Algoritmo 2 do apêndice A.

Note que a função que decripta Dec não usa o verificador randômico $\mathfrak{v}_{i,j}$. Por isto, não se faz necessário incluí-lo como parâmetro. Figura 1.12 esquematiza a comunicação de protocolos usando PCHA para proteger a privacidade. O agregador também poderia ser virtual, *i.e.*, a consolidação poderia ocorrer com os usuários enviando suas respectivas mensagens encriptadas $\mathfrak{M}_{i,j}$ uns para os outros de forma que um usuário tenha o produto de todas elas formando a consolidação encriptada $\mathfrak{C}_j$ e envie $\mathfrak{C}_j$ para o contador descriptografar.

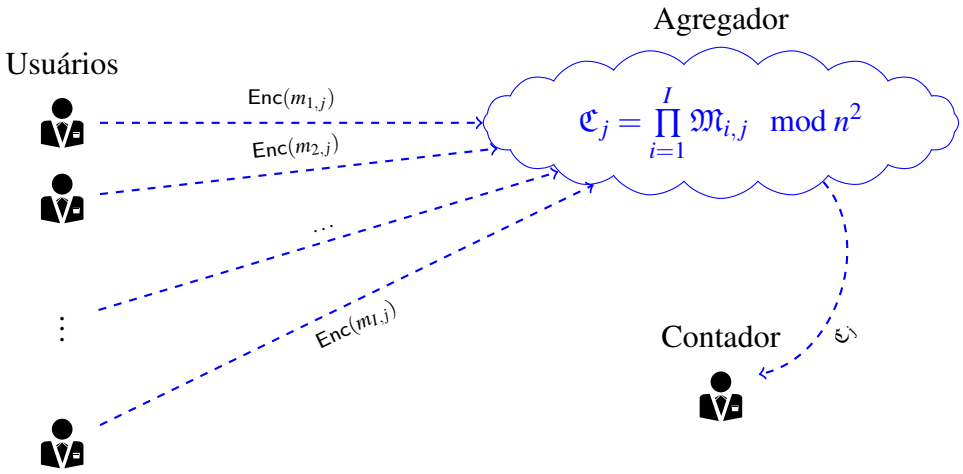

Figura 1.12: Modelo de comunicação para PCHA no tempo j.

Considerando a necessidade de um agregador, poderíamos construir uma SDC-Net em estrela com a propriedade de uma PCHA. Para medirmos e compararmos performance, SDC-Net em estrela apresenta um limite inferior e deve ser comparado com os novos protocolos [Borges de Oliveira 2017c]. Similarmente a SDC-Net, PCHA precisa de uma função de assinatura Sign. Em particular, pode-se usar alguma técnica de assinatura digital $\mathfrak{S}_{i,j}$ homomórfica, de forma que o agregador saiba quais usuários enviaram suas mensagens encriptadas $\mathfrak{M}_{i,j}$.

1.5.3. DC-Net Assimétrica

[Borges de Oliveira 2017g] apresenta o conceito de ADC-Net, que é similar ao de SDC-Net. Em particular, SDC-Net apresenta a possibilidade de verificação como técnicas de compromisso, e consequentemente, os usuários não podem romper o protocolo enviando um valor errado.

Em [Borges de Oliveira 2017g], um protocolo que protege a privacidade é definido como uma ADC-Net se satisfaz as seguintes propriedades:

1. o protocolo tem todas as propriedades de uma SDC-Net, excluindo segurança incondicional;
2. a segurança é baseada em uma função criptográfica *arapuca* do inglês *trapdoor*;
3. usuários podem usar chaves permanentes;
4. o tempo de processamento tem complexidade máxima polinomial;
5. não é necessário uma iteração sobre o número de usuários I, excluindo na consolidação;
6. usuários podem mandar o número mínimo de mensagens;
7. usuários podem usar uma função de assinatura para gerar uma assinatura digital $\mathfrak{S}_{i,j}$ de cada uma de suas mensagens $m_{i,j}$;
8. similar a uma técnica de comprometimento, usuários podem verificar suas mensagens $m_{i,j}$.

Baseado na propriedade 1, ADC-Net não precisa de um agregador. Dependendo do protocolo, ou todos podem descobrir o resultado da consolidação c_j, ou apenas um contador pode descobrir o resultado da consolidação c_j [Borges 2016]. Este capítulo usa basicamente a mesma ADC-Net apresentada em [Borges de Oliveira 2017g], com a diferença que todos os usuários enviam suas mensagens encriptadas para todos em vez de enviarem para um contador. Ambas os casos são equivalentes.

Durante o processo de inicialização do protocolo, os usuários escolhem um produto de primos n, por exemplo, como dado em [Boneh and Franklin 2001]. Cada usuário i escolhe uma chave privada k_i. Na sequência, eles determinam

$$s = \sum_{i=1}^{I} k_i,$$

de forma que todos saibam do valor de s sem revelar suas respectivas chaves privadas k_i, por exemplo, com uma SDC-Net.

Depois da configuração inicial, os usuários podem começar a cifrar suas mensagens com a função que encripta

$$\begin{aligned} &\mathsf{Enc} : \mathbb{Z}_n \to \mathbb{Z}_{n^2} \\ &\mathsf{Enc}_i(m_{i,j}) \mapsto (1+n)^{m_{i,j}} \cdot g^{h_j+k_i} \mod n^2, \end{aligned} \tag{4}$$

onde $h_j = \mathsf{H}(j)$ e H é uma função de hash s.t. se comporta como uma função de mão única e é resistente a colisões.

Os usuários i encriptam suas mensagens $m_{i,j}$ gerando mensagens encriptadas $\mathfrak{M}_{i,j}$ que são declaradas publicamente. Eles podem usar uma função de assinatura Sign para garantir a origem da mensagem. Caso algum usuário i não envie sua mensagem $m_{i,j}$ na janela de tempo j, todos sabem quem é o usuário i e podem requisitar que ele envie a respectiva mensagem encriptada $\mathfrak{M}_{i,j}$ com a assinatura digital $\mathfrak{S}_{i,j}$. No pior caso, os usuários podem inicializar novamente o protocolo excluindo quem não está enviando as mensagens encriptadas.

Para gerar a consolidação encriptada $\mathfrak{C}_j$ das mensagens encriptadas $\mathfrak{M}_{i,j}$, os usuários calculam

$$\mathfrak{C}_j = \prod_{i=1}^{I} \mathfrak{M}_{i,j} \mod n^2, \tag{5}$$

Note que o produtório que gera a consolidação encriptada $\mathfrak{C}_j$ pode ser calculado conforme as mensagens encriptadas $\mathfrak{M}_{i,j}$ vão chegando. Porém, o produtório somente resulta a consolidação encriptada $\mathfrak{C}_j$ após todas as mensagens encriptadas $\mathfrak{M}_{i,j}$ forem computadas. Consequentemente, a função que decripta Dec só pode ser executada após o final do produtório.

Para descriptografar a consolidação encriptada $\mathfrak{C}_j$, eles calculam

$$\begin{aligned} &\mathsf{Dec} : \mathbb{Z}_{n^2} \to \mathbb{Z}_n \\ &\mathsf{Dec}\left(\mathfrak{C}_j\right) \mapsto \frac{\left(\mathfrak{C}_j \cdot g^{-I \cdot h_j - s} \mod n^2\right) - 1}{n}, \end{aligned} \tag{6}$$

onde $s = \sum_{i=1}^{I} k_i$.

Pode-se mostrar que os protocolos usando Equações (4) a (6) geram uma ADC-Net, *i.e.*, satisfazem as oito propriedades descritas acima. É interessante notar que ADC-Nets são generalizações de PCHAs, *i.e.*, PCHAs são casos particulares de ADC-Nets. Pode-se criar ADC-Nets com equações que posam ser simplificadas de forma que resultem em PCHAs. Note também que Equação (4) poderia conter um fator randômico elevado a n, *i.e.*, $(\mathfrak{v}_{i,j})^n$, mas não se faz necessário.

O processo de construção de s poderia ser diferente. Em vez dos usuários gerarem s a partir de suas chaves privadas k_i, estas poderiam ser geradas para determinar um s fixo. Assim, ficaria fácil criar grupos de usuários confiáveis cuja soma das chaves dos membros de cada grupo de s. Para simplificar a função que decripta Dec na Equação (6), s poderia ser igual a n. O resultado das funções de hash H dos grupos poderiam ser diferentes, *i.e.*, existem várias formas de determinarmos grupos de usuários confiáveis em ADC-Net. Talvez a forma mais interessante seja quando a soma s é igual a zero para uma ADC-Net completa. Em particular, a soma s pode ser dada como na Figura 1.9 ou Figura 1.10.

Em técnicas de comprometimento, este capítulo descreve duas formas de verificação, a saber, consolidação comprometida do usuário i, *i.e.*, $\mathfrak{U}_i$ e consolidação comprometida do tempo j, *i.e.*, $\mathfrak{T}_j$. Com ADC-Nets, não precisamos verificar $\mathfrak{T}_j$, pois é possível descriptografar a consolidação encriptada $\mathfrak{C}_j$ e acessar a consolidação c_j no tempo j. Ao

descriptografar já temos a garantia que os valores estão comprometidos. Neste ponto, dependendo da aplicação, o protocolo que protege a privacidade pode verificar c_j com algum valor externo, *e.g.*, o número de votantes. Além disto, o usuário i pode fazer comprovações, *e.g.*, se ele votou nas últimas eleições. Para fazermos uma verificação sobre as mensagens do usuário i, calculamos

$$\mathfrak{U}_i = \prod_{j=1}^{J} \mathfrak{M}_{i,j}$$

e

$$\mathfrak{H} = \prod_{j=1}^{J} g^{h_j}$$

O usuário i calcula

$$\mathfrak{V}_i = \prod_{j=1}^{J} (1+n)^{m_{i,j}} \cdot g^{k_i} \mod n^2.$$

A função de abertura Open pode determina se os valores estão corretos, *i.e.*,

$$\mathsf{Open}\left(\mathfrak{U}_i, \mathfrak{V}_i, \mathfrak{H}\right) = \left(\mathfrak{U}_i \stackrel{?}{=} \mathfrak{V}_i \cdot \mathfrak{H}\right).$$

Se a função retornar verdadeiro, os valores estão corretos.

Diferente, poderíamos querer verificar o valor de consolidação por usuário b_i sem revelar as mensagens $m_{i,j}$ do usuário i. Desta forma, calculamos como anteriormente

$$\mathfrak{U}_i = \prod_{j=1}^{J} \mathfrak{M}_{i,j}.$$

Porém, o usuário i calcula

$$b_i = \sum_{j=1}^{J} m_{i,j}$$

e

$$\mathfrak{V}_i = \prod_{j=1}^{J} g^{h_j + k_i} \mod n^2.$$

A função de abertura Open pode tem que ser definida de forma diferente para determinar se os valores estão corretos, *i.e.*,

$$\mathsf{Open}\left(\mathfrak{U}_i, b_i, \right) = \left(\mathfrak{U}_i \stackrel{?}{=} (1+n)^{b_i} \cdot \mathfrak{V}_i\right).$$

A ADC-Net poderia ser construída para retornar o valor da consolidação por usuário b_i. No entanto, isto deve ser feito com muito cuidado para evitar o comprometimento do protocolo com o vazamento de informações privadas, *i.e.*, mensagens $m_{i,j}$.

Considerando verificações, seria mais aconselhável usar a função que encripta Enc dada por

$$\begin{aligned} \mathsf{Enc} &: \mathbb{Z}_n \to \mathbb{Z}_{n^2} \\ \mathsf{Enc}_i(m_{i,j}) &\mapsto (1+n)^{m_{i,j}} \cdot g^{h_j \cdot k_i} \mod n^2, \end{aligned}$$

em vez de Equação (4). A diferença está no produto dos expoentes, em vez da soma.

Logo, a respectiva função inversa, *i.e.*, a função que decripta Dec é dada por

$$\begin{aligned} &\mathsf{Dec} : \mathbb{Z}_{n^2} \to \mathbb{Z}_n \\ &\mathsf{Dec}\left(\mathfrak{C}_j\right) \mapsto \frac{\left(\mathfrak{C}_j \cdot g^{-h_j \cdot s} \mod n^2\right) - 1}{n}, \end{aligned}$$

onde $s = \sum_{i=1}^{I} k_i$.

Similarmente, poderíamos construir diversas ADC-Nets, *e.g.*,

$$\begin{aligned} &\mathsf{Enc} : \mathbb{Z}_n \to \mathbb{Z}_{n^2} \\ &\mathsf{Enc}(m_{i,j}) \mapsto (1+n)^{m_{i,j}} \cdot h_j^{k_i} \mod n^2 \end{aligned}$$

e

$$\begin{aligned} &\mathsf{Dec} : \mathbb{Z}_{n^2} \to \mathbb{Z}_n \\ &\mathsf{Dec}\left(\mathfrak{C}_j\right) \mapsto \frac{\left(\mathfrak{C}_j \cdot h_j^{-s} \mod n^2\right) - 1}{n}. \end{aligned}$$

Note que um número de usuários poderia se juntar para provar alguma propriedade do conjunto sem que as mensagens $m_{i,j}$ individuais deles sejam divulgadas. Similarmente, poderiam detectar um usuário querendo romper o protocolo que protege a privacidade. Por exemplo, suponha que um usuário i queira romper o protocolo, desta forma, ele poderia enviar um valor que preencha todos os bits de sua mensagem $m_{i,j}$ no tempo j. Se é feito uma consolidação c_j no tempo j e uma consolidação por usuário b_i por usuário i, então construir uma tabela com as consolidações e descobrir exatamente a mensagem $m_{i,j}$ que foi comprometida. A detecção do rompimento pode ser visualizada na Tabela 1.2. Mesmo que o usuário i enviasse várias mensagens comprometidas, eles seriam detectados com a Tabela 1.2.

O problema acontece quando não há duas consolidações ou quando não se deseja esperar a segunda consolidação para detectar a origem das mensagens que podem romper o protocolo. Outro problema acontece quando o usuário i envia uma mensagem $m_{i,j}$ pequena, mas que é inválida. Por exemplo, resultando que a soma dos votos seja maior que o número de votantes. Neste caso, pode-se separar os usuários em dois conjuntos $\mathscr{U}_1$ e $\mathscr{U}_2$. Na sequência, pode-se pedir aos usuários de cada conjunto para verificarem suas mensagens. Um dos dois grupos está com a soma errada resultando mais votantes que votos. O conjunto com erro pode ser dividido novamente, e para não comprometermos a privacidade, os usuários do conjunto não comprometido podem se juntar ao novo conjunto para ajudar na proteção da privacidade. Com este processo, podemos detectar o usuário i que tenta romper o protocolo em $\log_2(I)$ passos. Desta forma, temos

$$v = \sum_{i \in \mathscr{U}_1} m_{i,j}, \tag{7}$$

e

$$\mathfrak{V} = \prod_{i \in \mathscr{U}_1} g^{h_j + k_i} \mod n^2. \tag{8}$$

Tabela 1.2: Detectando a mensagem $m_{i,j}$ com um valor gigante.

	1	2	$\cdots$	j	$\cdots$	J	b_i
1	$m_{1,1}$	$m_{1,2}$	$\cdots$	$m_{1,j}$	$\cdots$	$m_{1,J}$	$\sum_{j=1}^{J} m_{1,j}$
2	$m_{2,1}$	$m_{2,2}$	$\cdots$	$m_{2,j}$	$\cdots$	$m_{2,J}$	$\sum_{j=1}^{J} m_{2,j}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
i	$m_{i,1}$	$m_{i,2}$	$\cdots$	$m_{i,j}$	$\cdots$	$m_{i,J}$	$\sum_{j=1}^{J} m_{i,j}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
I	$m_{I,1}$	$m_{I,2}$	$\cdots$	$m_{I,j}$	$\cdots$	$m_{I,J}$	$\sum_{j=1}^{J} m_{I,j}$
c_j	$\sum_{i=1}^{I} m_{i,1}$	$\sum_{i=1}^{I} m_{i,2}$	$\cdots$	$\sum_{i=1}^{I} m_{i,j}$	$\cdots$	$\sum_{i=1}^{I} m_{i,J}$	$\sum_{j=1}^{j} c_j = \sum_{i=1}^{I} b_i$

O contador ou um interessando na verificação calcula

$$\mathfrak{P} = \prod_{i \in \mathscr{U}_1} \mathfrak{M}_{i,j} \tag{9}$$

Logo a função de abertura Open é definida por

$$\mathsf{Open}\left(\mathfrak{P}, \mathfrak{V}, v\right) = \left(\mathfrak{P} \stackrel{?}{=} (1+n)^v \cdot \mathfrak{V} \mod n^2\right). \tag{10}$$

O usuário que tenta romper o protocolo não tem escapatória, se ele está no conjunto $\mathscr{U}_1$, ou o valor de v não é o esperado, ou a função de abertura Open não vai abrir o comprometimento. Como ou o usuário está no conjunto $\mathscr{U}_2$ ou está no conjunto $\mathscr{U}_1$, consequentemente, se o conjunto $\mathscr{U}_1$ não tem problema, então ele está no conjunto $\mathscr{U}_2$.

Sem perda de generalidade, suponha que o problema está no $\mathscr{U}_1$, logo podemos separar os usuários do conjunto $\mathscr{U}_1$ em dois conjuntos $\mathscr{U}_{1_1}$ e $\mathscr{U}_{1_2}$, tal que

$$\mathscr{U}_{1_1} \cup \mathscr{U}_{1_2} = \mathscr{U}_1.$$

Para não comprometermos a privacidade podemos espalhar os usuários do $\mathscr{U}_2$ nos conjuntos $\mathscr{U}_{1_1}$ e $\mathscr{U}_{1_2}$, assim temos

$$\mathscr{U}_{1_1} \cup \mathscr{U}_{1_2} = \mathscr{U}_1 \cup \mathscr{U}_2.$$

Finalmente, calcula-se de forma semelhante as Equações (7) a (10), trocando apenas os conjuntos. Apesar de não se ter mais controle sobre o valor de v, tem-se o controle de quais usuários podem estar tentando romper o protocolo. O processo recursivo leva a detecção do usuário em $\log_2(I)$ passos.

1.6. Comparações

Esta seção apresenta uma comparação entre as melhores técnicas, enfatizando as semelhanças e diferenças, ou seja, lista-se qual técnica tem ou não uma propriedade. Por exemplo, que técnica garante verificação, ou garante que ninguém vai romper a privacidade, qual técnica é livre de terceiros confiáveis, etc. Especificamente, ADC-Nets podem garantir verificação de forma semelhante a commitment e ainda podem decriptar o resultado total das operações homomórficas, ou seja, computadas sobre os valores encriptados. Um outro ponto avaliado é a performance das técnicas apresentadas. Certamente, algoritmos que podem rodar com pouco processamento são fundamentais para equipamentos com restrições de hardware. No entanto, para computação ubíqua, usar menos processamento em bilhões de dispositivos significa economizar energia e produzir dispositivos com custo mais acessível. Nesta seção, pode-se ver que ADC-Nets superam as outras técnicas em performance além de proverem mais propriedades. Para isto, listamos a complexidade dos algoritmos na Tabela 1.3, considerando que o sistema criptográfico de Paillier tem o melhor desempenho das PCHA. Além do valor do número de usuários I e número de tempo J, a Tabela 1.3 contém n que é o produto de primos com no mínimo 1024 bits e k que é as chaves secretas de no mínimo 180 bits escolhidas pelos usuários. SDC-Net pode ser o mais rápido para um pequeno número de usuários I. Porém, vai ficando lento conforme o número de usuários I cresce. Técnicas de compromisso podem fazer verificações no número de usuários I e número de tempo J.

Tabela 1.3: Comparação da complexidade computacional.

Técnica	Enc	Consolidação	Dec
SDC-Net	$O(I)$	NA	$O(I)$
Compromisso	$O(\log(k))$	$O(J)$ ou $O(I)$	$O(k)$
PCHA	$O(\log(n))$	$O(I)$	$O(\log(n))$
ADC-Net	$O(\log(k))$	$O(I)$	$O(\log(k))$

Considerando os valores de n e k vemos que protocolos baseados em ADC-Net tendem a ser bem mais rápidos que protocolos baseados em PCHA. Acima de tudo, k cresce muito mais devagar que n quando se aumenta o nível de segurança. O número de bits das chaves para um dado nível de segurança é apresentado na Tabela 1.4. As diferenças são equivalentes as técnicas baseadas em curvas elípticas comparadas com o problema da fatoração de inteiros. Por isto, aparentemente não vale a pena construir uma ADC-Net baseada em curvas elípticas. Para uma comparação analítica da performance veja [Borges de Oliveira 2017a], e para uma comparação através de simulação, veja [Borges de Oliveira 2017i]. A Tabela 1.3 contém não aplicável (NA) na consolidação encriptada com SDC-Net porque o processo de consolidação acontece junto com a função que decripta Dec.

Podemos ver na Figura 1.13 que o número de bits de n tem um crescimento exponencial quando aumentamos o nível de segurança. Enquanto que o número de bits de k tem um crescimento linear com o aumento do nível de segurança. Uma vez que, a interpolação dos pontos da Tabela 1.4 nos fornece as curvas $y = 2x$ para os valores de k e $y = 601.53\exp(0.0127087x)$ para os valores de n. O tamanho da chave é diretamente pro-

Tabela 1.4: Comparação entre o crescimento de k e n com o nível de segurança.

Força Bruta nível de segurança	k	n
80	160	1 024
112	224	2 048
128	256	3 072
192	384	7 680
256	512	15 360

porcional ao custo computacional. Portanto, ADC-Net é cada vez mais rápida que PCHA quando o nível de segurança aumenta.

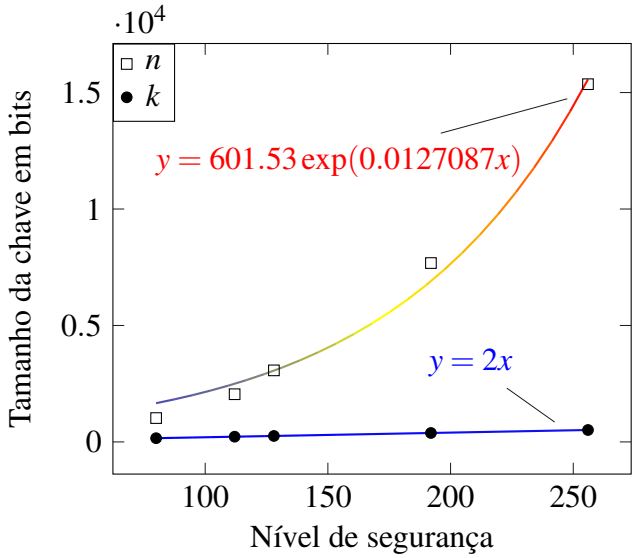

Figura 1.13: Curvas do crescimento de k e n em função do nível de segurança.

Algoritmos para SDC-Net tem diversas propriedades, ADC-Net tem mais ainda. Algoritmos de comprometimento podem ser completamente substituídos por algoritmos de ADC-Net. Neste caso, podemos comparar apenas as propriedades de SDC-Net, PCHA e ADC-Net. Uma propriedade importante é a capacidade de evitar conluio. Usando DC-Nets, podemos garantir que só se vaza as mensagens de um usuário quando todos estão contra ele, *i.e.*, $I-1$ usuários devem conspirar para revelarem mensagens de um usuário. No caso de PCHA, basta que um agregador envie mensagens encriptadas $\mathfrak{M}_{i,j}$ para um contador que possui a chave privada, e consequentemente, pode descriptografar as mensagens encriptadas $\mathfrak{M}_{i,j}$ que recebe. Logo basta o conluio de duas entidades. O valor de $I-1$ é o melhor possível em privacidade. DC-Nets tem o conceito de conjunto de usuários confiáveis, garantindo que mensagens encriptadas $\mathfrak{M}_{i,j}$ do mesmo conjunto de usuários somente serão descriptografadas juntas. Com os usuários confiáveis, o conluio de todos os outros não é suficiente para vazamento de informação, mais ainda, pode-se diminuir o tempo de processamento para SDC-Net.

Todas as três técnicas podem e devem usar uma função de assinatura Sign para ge-

rar uma assinatura digital $\mathfrak{S}_{i,j}$ de cada mensagem encriptada $\mathfrak{M}_{i,j}$. Porém, apenas usando DC-Nets, o destinatário pode verificar quem enviou ou não e se a assinatura digital $\mathfrak{S}_{i,j}$ está correta, pois ADC-Net necessita de um agregador. Consequentemente, os usuários podem descriptografar a consolidação c_j com DC-Nets, mas não podem com PCHA.

Pode-se construir protocolos que protejam a privacidade que enviam o mínimo número de mensagens, mas o tempo de processamento não é constante em todas as técnicas. Conforme o número de usuários I cresce, protocolos baseados em SDC-Net ficam lentos, logo eles não são escaláveis com o número de usuários. Quanto ao processo da consolidação encriptada $\mathfrak{C}_j$ também depender do número de usuários I, esta faz operações bem mais simples e cresce linearmente com o número de usuários I, já função que encripta Enc cresce quadraticamente no total, apesar de linearmente para cada usuário.

Todas as técnicas permitem o uso de chaves permanentes, *i.e.*, chaves que podem ser geradas em um processo inicial e usadas para gerar várias mensagens encriptadas $\mathfrak{M}_{i,j}$. Porém, o número de chaves varia de acordo com a técnica. Cada usuário em uma SDC-Net completa precisa armazenar $2(I-1)$ chaves, mas o total de chaves no protocolo cresce quadraticamente $O(I^2)$ com o número de usuários I. Já usando PCHA, cada i tem uma chave criptográfica e existem apenas duas chaves criptográficas no protocolo, a saber, a pública que encripta e a privada que decripta. Com ADC-Net a situação é diferente. Cada usuário tem uma chave privada única, mas o número total de chaves no protocolo cresce linearmente com o número de usuários I. Apesar de protocolos baseados em ADC-Net terem mais chaves que protocolos baseados em PCHA, eles têm o menor número de chaves para garantir a privacidade e evitar o conluio. Usando ADC-Net, cada usuário tem sua chave privada e a chave inversa para descriptografar pode ser pública ou privada.

Pode-se projetar protocolos que rodem em tempo polinomial com as três técnicas. Porém, somente ADC-Net possibilita verificação de mensagens encriptadas $\mathfrak{M}_{i,j}$ sem violar a privacidade dos usuários. Consequentemente, nenhum usuário i pode tentar romper o protocolo enviando mensagens $m_{i,j}$ erradas que ele será detectado. Em teoria, técnicas de PCHA poderiam possibilitar usuários a verificar se enviaram a mensagem encriptada $\mathfrak{M}_{i,j}$ correta, mas quem possui a chave privada não poderia verificar, pois não poderia receber as mensagens $m_{i,j}$ sem possibilidade de acessar as mensagens $m_{i,j}$.

Excluindo a complexidade computacional já apresentada na Tabela 1.3, apresenta-se na Tabela 1.5 um resumo das propriedades discutidas nesta seção.

1.7. Considerações Finais

Esta seção enfatiza os pontos principais das seções anteriores junto com considerações sobre as técnicas apresentadas neste capítulo. Acima de tudo, esta seção apresenta as limitações encontradas nas técnicas com suas primitivas criptográficas, os desafios encontrados nos cenários e uma perspectiva de futuros trabalhos voltando a discutir os cenários práticos de aplicação de proteção à privacidade.

Seção 1.1 tem apresentado uma visão geral deste capítulo começando a introduzir a necessidade de preservarmos a privacidade, enquanto que a Seção 1.2 tem apresentado diversos cenários onde se faz necessário a preservação da privacidade. Seção 1.3 tem

Tabela 1.5: Comparação das propriedades.

Propriedades	**SDC-Net**	**PCHA**	**ADC-Net**
Evita conluio	✓		✓
Conjunto de usuários confiáveis	✓		✓
Mensagens direto para o destinatário	✓		✓
Usuários podem descriptografar	✓		✓
Número mínimo de mensagens	✓	✓	✓
Escalável		✓	✓
Chaves permanentes	✓	✓	✓
Baseado em *trapdoors*	✓	✓	✓
Chaves armazenadas por usuário	$2(I-1)$	1	1
Total de chaves	$O(I^2)$	2	$O(I)$
Tempo polinomial	✓	✓	✓
Possibilidade de verificação			✓
Não se pode romper o protocolo			✓

apresentado como podemos avaliar as técnicas usadas para proteger a privacidade. Em particular, uma análise meramente das técnicas não apresenta qual a probabilidade de cada mensagem. Considerando que as melhores técnicas devem deixar as mensagens equiprováveis, devemos analisar a probabilidade de se inferir as mensagens com os dados que podem ser encontrados no problema de cada cenário. Seção 1.4 tem apresentado as técnicas simétricas para proteger a privacidade, enquanto que a Seção 1.5 tem apresentado as técnicas assimétricas. A comparação entre os pontos mais importantes das técnicas é apresentada na Seção 1.6.

SDC-Net é a técnica simétrica que apresenta mais propriedades, logo a mais interessante. Entre elas, ADC-Net pode garantir segurança incondicional, mas os usuários só podem usar suas chaves uma vez. Podemos usar a chaves várias vezes usando uma função de hash, que apesar de ser rápida, o tempo de processamento para encriptar usando SDC-Net cresce quadraticamente com o número de usuários.

As técnicas assimétricas têm um tempo de processamento independentemente do número de usuários. Além disto, cada usuário tem apenas uma chave. Usando PCHA, todos têm a mesma chave pública, mas somente uma entidade tem a chave privada que pode decifrar todas as mensagens. Por esta razão, tal entidade não pode ter acesso ao processo de consolidação encriptada. Ainda mais, ninguém que tenha acesso ao processo pode criar um conluio com tal entidade. Portanto, os usuários não têm garantia de privacidade, *i.e.*, que suas mensagens encriptadas não serão decriptadas. Diferentemente, os usuários têm garantia que suas mensagens encriptadas não serão decifradas individualmente quando eles usam DC-Nets. Além desta propriedade, ADC-Net tem todas as propriedades de SDC-Net. Em adição, ADC-Net também possibilita verificações de forma semelhante que em técnicas de comprometimento, e consequentemente, os usuários não podem romper o protocolo enviando uma mensagem inválida. ADC-Net força a garantia da privacidade e possibilita auditorias mantendo a privacidade.

A fase de inicialização com PCHA é mais simples do que com ADC-Net. Com a

primeira técnica, todos os usuários recebem a mesma chave pública de quem tem a chave privada. Porém, isto gera insegurança. Com a segunda técnica, todos os usuários devem escolher suas respectivas chaves privadas e juntos gerarem a chave inversa que pode ser pública ou privada, mas que decripta apenas a consolidação encriptada, *i.e.*, mensagens individuais não podem ser decriptadas. A fase de inicialização de protocolos com ADC-Net poderia ser tão simples como com PCHA se for introduzido uma autoridade confiável para distribuir chaves. Porém, tal autoridade seria um ponto crítico de falha. Assim, como a entidade que detém a chave privada de uma técnica de PCHA é um ponto crítico de falha.

A inserção e remoção de usuários nos protocolos é bem mais simples com PCHA do que com DC-Nets. Basta enviar a chave pública e revogar o reconhecimento da assinatura digital para técnicas de PCHA. Para o caso de DC-Nets, os protocolos devem ser reinicializados. Aqui existe uma relação inversa entre praticidade e privacidade.

Diversos cenários onde se deve preservar a privacidade precisam da operação de soma, *e.g.*, votação eletrônica, sistemas de reputação, redes de sensores, cibermedicina, processamento de imagens, dinheiro eletrônico, computação em múltiplas partes, privacidade no mundo acadêmico, redes inteligentes *etc.* Em cibermedicina, encontramos operações com textos, por exemplo, no prontuário eletrônico. DC-Nets também podem trabalhar com textos criando uma rede de anonimato. No entanto, buscas em textos encriptados e encriptação com múltiplas chaves têm maiores aplicações em textos que somas. Acima de tudo, devemos considerar que se protegermos uma operação de soma em qualquer cenário, então estaremos protegendo todas as operações subsequentes. Este é o processo de proteção de alguns protocolos de proteção da privacidade em processamento de imagens.

A definição do problema é um dos desafios encontrados nos cenários para proteção da privacidade. Por exemplo, se tentarmos proteger um dado médico, teremos que proteger todos os lugares onde ele aparece, mas nem sempre está claro onde este dado aparece. Mais ainda, a relação entre outros dados pode levar a dedução do dado protegido. Todos os cenários parecem ter uma relação inversa entre privacidade e disponibilidade da informação. No entanto, esta relação inversa não necessariamente existe. Muitos cenários podem ter a privacidade protegida com a informação disponível no momento certo. Quando se atinge a proteção almejada, temos que começar os processos de otimização para reduzir custos de processamento.

Protocolos baseados em ADC-Net podem requisitar menor espaço para armazenamento das chaves e serem cada vez mais rápidos que PCHA quando o nível de segurança aumenta. Aparentemente, não existe vantagem em desenvolver ADC-Net baseadas em outras primitivas, *e.g.*, curvas elípticas. O tempo de processamento ótimo para encriptar mensagens para gerar uma consolidação é alcançado com uma SDC-Net estrela, *i.e.*, $O(1)$. Portanto, quaisquer novas técnicas deveriam ter a performance comparada com uma SDC-Net estrela e o estado da arte de ADC-Net.

O uso de algoritmos criptográficos para proteger a privacidade é relativamente novo. Em particular, SDC-Net foi criada quase trinta anos antes da recém-criada ADC-Net. Todos os cenários onde se deseja proteger a privacidade devem ser revisitados com novas ferramentas, em especial com ADC-Net.

Como tecnologias da informação e comunicação estão continuamente conectando pessoas, dados e dispositivos, temos cada dia mais um maior volume de dados para tratar e informações privadas para proteger. O vazamento de tais informações tem impacto direto na vida de cada cidadão. Muitos problemas estão em aberto ou precisam ser otimizados. Faz-se necessário muita pesquisa na área de privacidade.

A. Algoritmos

Algorithm 1: Multi-exponenciação modular

Input: Inteiros b_i, e_i, m, n s.t. $e_i = \sum_{j=1}^{l_i} 2^{j-1} e_{i_j}$, onde $l_i = \lceil \log_2 e_i \rceil$ e $e_{i_j} \in \{0,1\}$.

Output: $\prod_{i=1}^{n} b_i^{e_i} \mod m$.

```
1 L ← ⌈max(log2 e1, ..., log2 en)⌉
2 a ← 1
3 for j = L to 1 by −1 do
4     a ← a^2 mod m
5     for i = 1 to n do
6         if e_{i_j} = 1 then
7             a ← a · b_i mod m
8 return a
```

Algorithm 2: Inversa multiplicativa do elemento a de um grupo $\mathbb{Z}_n$ usando o Algoritmo Euclidiano Estendido

Input: Inteiros a e n
Output: $a^{-1} \mod n$

$a \leftarrow a \mod n$
$t \leftarrow 0$
$t' \leftarrow 1$
$r \leftarrow n$
$r' \leftarrow a$
while $r' \neq 0$ **do**
  $q \leftarrow \lfloor r/r' \rfloor$
  $t \leftarrow t'$
  $t' \leftarrow t - q \cdot t'$
  $r \leftarrow r'$
  $r' \leftarrow r - q \cdot r'$
if $r > 1$ **then**
  return "a não tem inversa"
if $t < 0$ **then**
  $t \leftarrow t + n$
return t

B. Lista de Acrônimos

ADC-Net DC-Net assimétrica. 21, 27–30, 32–36

DC-Net *Dining Cryptographers Network*. 16, 33–36

NA não aplicável. 32

PCHA primitiva de criptografia homomórfica aditiva. 12, 13, 20, 25, 26, 28, 32–36
PMU *phasor measurement unit*. 12

SDC-Net DC-Net simétrica. 16, 18–21, 23–27, 32–36

C. Lista de Abreviações

e.g. "por exemplo" de *exempli gratia* em Latim. 3–5, 7, 16–19, 22, 25, 29, 30, 36, 40
etc. "e outros" ou "e assim por diante" de *et cetera* em Latim. 4, 6, 7, 13, 36, 40

i.e. "isto é" ou "ou seja" de *id est* em Latim. 2, 5, 6, 8, 10, 12–15, 17–20, 22, 24–26, 28–30, 33–36, 39, 40
iff condição necessária e suficiente de "if and only if" em Inglês. 39

s.t. "tal que" de *such that* em Inglês. 24, 25, 28, 37, 39

v. "contra" ou "em contraste com" de *versus* em Latim. 4

D. Lista de Símbolos

assinatura digital ($\mathfrak{S}_{i,j}$) assinatura digital do usuário i no tempo j. 23, 26–28, 34, 36, 39
atribuição ($\leftarrow$) $a \leftarrow a+1$ significa a atribuição de $a+1$ para a. 25, 37–39

caso ($\stackrel{?}{=}$) os valores são corretos quando a equação é satisfeita. 24, 25, 29, 31
consolidação (c_j) consolidação das mensagens agregadas no tempo j, *i.e.*, $c_j = \mathsf{Dec}\left(\mathfrak{C}_j\right)$. 12, 13, 20, 22–32, 34, 36, 39
consolidação comprometida do usuário i ($\mathfrak{U}_i$) consolidação das mensagens comprometidas no número de tempo J, *i.e.*, $\mathfrak{U}_i = \prod_{j=1}^{J} \mathfrak{N}_{i,j}$. 22, 24, 25, 28, 29, 39
consolidação comprometida do tempo j ($\mathfrak{T}_j$) consolidação das mensagens comprometidas do número de usuários I, *i.e.*, $\mathfrak{T}_j = \prod_{i=1}^{I} \mathfrak{N}_{i,j}$. 22–25, 28, 39
consolidação encriptada ($\mathfrak{C}_j$) consolidação encriptada das mensagens no tempo j, s.t. $\mathfrak{C}_j = \prod_{j=1}^{J} \mathfrak{M}_{i,j}$. 12, 13, 26, 28, 30, 32, 34–36, 39
consolidação por usuário (b_i) consolidação das mensagens comprometidas pelo do usuário i, *i.e.*, $b_i = \sum_{j=1}^{J} m_{i,j}$. 22, 24, 25, 29–31

função de abertura (Open) a função que abre mensagem comprometida $\mathfrak{N}_{i,j}$ e retorna verdade **iff** os valores estão corretos. 22–25, 29, 31
função de assinatura (Sign) função que retorna uma assinatura digital $\mathfrak{S}_{i,j}$. 23, 26–28, 33
função de comprometimento (Commit) um esquema de comprometimento definido de acordo com o protocolo. 22–24, 39
função de hash (H) uma função de hash s.t. se comporta como uma função de mão única e é resistente a colisões. 19, 20, 28, 35
função que decripta (Dec) uma função que decripta definida de acordo com o protocolo. 11–13, 26, 28, 30, 32, 39
função que encripta (Enc) uma função que encripta definida de acordo como protocolo. 11–13, 25–27, 29, 30, 32, 34, 39

inteiros ($\mathbb{Z}$) o conjunto dos números inteiros. 23–30, 38

mensagem ($m_{i,j}$) abstração de um dados ou informação do usuário i no tempo j. 3, 10–13, 15–31, 33–36, 39, 40
mensagem comprometida ($\mathfrak{N}_{i,j}$) mensagem comprometida de $m_{i,j}$, *i.e.*, $\mathfrak{N}_{i,j} = \mathsf{Commit}\left(m_{i,j}, \mathfrak{v}_{i,j}\right)$. 22–24, 39
mensagem encriptada ($\mathfrak{M}_{i,j}$) mensagem cifrada do usuário i no tempo j, *i.e.*, $\mathfrak{M}_{i,j} = \mathsf{Enc}\left(m_{i,j}\right)$. 11–13, 16, 17, 20, 25–29, 31, 33–35, 39

mínimo múltiplo comum (mmc) função que retorna o mínimo múltiplo comum. 25

número de tempo (J) número total de tempo j. 12, 22, 24, 29, 31, 32, 39, 40
número de usuários (I) número total de usuários i. 12, 13, 20–28, 30–35, 39, 40

tempo (j) identificação do tempo. 11, 12, 17, 19, 20, 22–32, 39, 40

usuário (i) abstração de um sistema computacional e seu respectivo usuário com identificação i. 3–5, 10–12, 16–36, 39, 40

verificador do usuário i ($\mathfrak{V}_i$) consolidação das mensagens comprometidas no número de tempo J, *i.e.*, $\mathfrak{V}_i = \prod_{j=1}^{J} \mathfrak{v}_{i,j}$. 22, 24, 25, 29, 40
verificador no tempo j ($\mathfrak{R}_j$) consolidação das mensagens comprometidas do número de usuários I, *i.e.*, $\mathfrak{R}_j = \prod_{i=1}^{I} \mathfrak{v}_{i,j}$. 23, 25, 40
verificador randômico ($\mathfrak{v}_{i,j}$) verificador aleatório da mensagem do usuário i no tempo j. 22–26, 28, 39, 40

E. Glossário

adversário uma abstração de entidades adversárias, atacantes, criminoso, *etc.* que tenta descobrir a mensagem $m_{i,j}$. 4–7, 11–13, 15–20

destinatário uma abstração da entidade que recebe a mensagem $m_{i,j}$, *e.g.*, receptor, destino, sumidouro, consumidor, *etc.* em geral um usuário. 11–13, 17, 34, 35

função de mão única se existe, é uma função que pode ser computada em tempo polinomial, mas sua inversa não pode. 28, 39

remetente uma abstração da entidade que envia a mensagem $m_{i,j}$, *e.g.*, emissor, origem, fonte, produtor, *etc.* em geral um usuário. 12, 13, 17

Referências

[Al Ameen et al. 2012] Al Ameen, M., Liu, J., and Kwak, K. (2012). Security and privacy issues in wireless sensor networks for healthcare applications. *Journal of Medical Systems*, 36(1):93–101.

[Bellare et al. 2007] Bellare, M., Boldyreva, A., and O'Neill, A. (2007). *Deterministic and Efficiently Searchable Encryption*, pages 535–552. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Boneh and Franklin 2001] Boneh, D. and Franklin, M. (2001). Efficient generation of shared rsa keys. *J. ACM*, 48(4):702–722.

[Borges 2016] Borges, F. (2016). *Privacy-Preserving Data Aggregation in Smart Metering Systems*. Energy Engineering Series. Institution of Engineering & Technology.

[Borges et al. 2017] Borges, F., Lara, P., and Portugal, R. (2017). Parallel algorithms for modular multi-exponentiation. *Applied Mathematics and Computation*, 292:406 – 416.

[Borges et al. 2012] Borges, F., Martucci, L. A., and Mühlhäuser, M. (2012). Analysis of privacy-enhancing protocols based on anonymity networks. In *Smart Grid Communications (SmartGridComm), 2012 IEEE Third International Conference on*, pages 378–383.

[Borges de Oliveira 2016] Borges de Oliveira, F. (2016). *On Privacy-Preserving Protocols for Smart Metering Systems: Security and Privacy in Smart Grids*. Springer International Publishing.

[Borges de Oliveira 2017a] Borges de Oliveira, F. (2017a). *Analytical Comparison*, pages 101–110. Springer International Publishing, Cham.

[Borges de Oliveira 2017b] Borges de Oliveira, F. (2017b). *Background and Models*, pages 13–23. Springer International Publishing, Cham.

[Borges de Oliveira 2017c] Borges de Oliveira, F. (2017c). *Concluding Remarks*, pages 127–129. Springer International Publishing, Cham.

[Borges de Oliveira 2017d] Borges de Oliveira, F. (2017d). *Introduction*, pages 3–12. Springer International Publishing, Cham.

[Borges de Oliveira 2017e] Borges de Oliveira, F. (2017e). *Quantifying the Aggregation Size*, pages 49–60. Springer International Publishing, Cham.

[Borges de Oliveira 2017f] Borges de Oliveira, F. (2017f). *Reasons to Measure Frequently and Their Requirements*, pages 39–47. Springer International Publishing, Cham.

[Borges de Oliveira 2017g] Borges de Oliveira, F. (2017g). *Selected Privacy-Preserving Protocols*, pages 61–100. Springer International Publishing, Cham.

[Borges de Oliveira 2017h] Borges de Oliveira, F. (2017h). *A Selective Review*, pages 25–36. Springer International Publishing, Cham.

[Borges de Oliveira 2017i] Borges de Oliveira, F. (2017i). *Simulation and Validation*, pages 111–126. Springer International Publishing, Cham.

[Camenisch et al. 2007] Camenisch, J., Lysyanskaya, A., and Meyerovich, M. (2007). Endorsed e-cash. In *Security and Privacy, 2007. SP '07. IEEE Symposium on*, pages 101–115.

[Chan and Perrig 2003] Chan, H. and Perrig, A. (2003). Security and privacy in sensor networks. *Computer*, 36(10):103–105.

[Chaum 1988] Chaum, D. (1988). The dining cryptographers problem: Unconditional sender and recipient untraceability. *J. Cryptol.*, 1(1):65–75.

[Chaum 1981] Chaum, D. L. (1981). Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24(2):84–90.

[Cramer et al. 2001] Cramer, R., Damgård, I., and Nielsen, J. B. (2001). Multiparty computation from threshold homomorphic encryption. In *Proceedings of the International Conference on the Theory and Application of Cryptographic Techniques: Advances in Cryptology*, EUROCRYPT '01, pages 280–299, London, UK, UK. Springer-Verlag.

[Cramer et al. 1997] Cramer, R., Gennaro, R., and Schoenmakers, B. (1997). A secure and optimally efficient multi-authority election scheme. In *Proceedings of the 16th Annual International Conference on Theory and Application of Cryptographic Techniques*, EUROCRYPT'97, pages 103–118, Berlin, Heidelberg. Springer-Verlag.

[De Montjoye et al. 2013] De Montjoye, Y.-A., Hidalgo, C. A., Verleysen, M., and Blondel, V. D. (2013). Unique in the crowd: The privacy bounds of human mobility. *Scientific reports*, 3.

[Díaz et al. 2003] Díaz, C., Seys, S., Claessens, J., and Preneel, B. (2003). Towards measuring anonymity. In *Proceedings of the 2Nd International Conference on Privacy Enhancing Technologies*, PET'02, pages 54–68, Berlin, Heidelberg. Springer-Verlag.

[Dwork 2008] Dwork, C. (2008). Differential privacy: A survey of results. In Agrawal, M., Du, D., Duan, Z., and Li, A., editors, *Theory and Applications of Models of Computation*, volume 4978 of *Lecture Notes in Computer Science*, pages 1–19. Springer Berlin Heidelberg.

[El Gamal 1985] El Gamal, T. (1985). A public key cryptosystem and a signature scheme based on discrete logarithms. In *Proceedings of CRYPTO 84 on Advances in Cryptology*, pages 10–18, New York, NY, USA. Springer-Verlag New York, Inc.

[Farhi et al. 2012] Farhi, E., Gosset, D., Hassidim, A., Lutomirski, A., and Shor, P. (2012). Quantum money from knots. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12, pages 276–289, New York, NY, USA. ACM.

[Gritzalis 2002] Gritzalis, D. A. (2002). Principles and requirements for a secure e-voting system. *Computers & Security*, 21(6):539 – 556.

[Jøsang et al. 2007] Jøsang, A., Ismail, R., and Boyd, C. (2007). A survey of trust and reputation systems for online service provision. *Decision Support Systems*, 43(2):618 – 644. Emerging Issues in Collaborative Commerce.

[Kerschbaum 2009] Kerschbaum, F. (2009). A verifiable, centralized, coercion-free reputation system. In *Proceedings of the 8th ACM Workshop on Privacy in the Electronic Society*, WPES '09, pages 61–70, New York, NY, USA. ACM.

[Li et al. 2007] Li, N., Li, T., and Venkatasubramanian, S. (2007). t-closeness: Privacy beyond k-anonymity and l-diversity. In *2007 IEEE 23rd International Conference on Data Engineering*, pages 106–115.

[Li and Cao 2013] Li, Q. and Cao, G. (2013). Efficient privacy-preserving stream aggregation in mobile sensing with low aggregation error. In De Cristofaro, E. and Wright,

M., editors, *Privacy Enhancing Technologies*, volume 7981 of *Lecture Notes in Computer Science*, pages 60–81. Springer Berlin Heidelberg.

[Naor and Shamir 1995] Naor, M. and Shamir, A. (1995). *Visual cryptography*, pages 1–12. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Paillier 1999] Paillier, P. (1999). Public-key cryptosystems based on composite degree residuosity classes. In *Advances in Cryptology - EUROCRYPT 1999*, volume 1592 of *Lecture Notes in Computer Science*, pages 223–238. Springer.

[Pedersen 1992] Pedersen, T. P. (1992). Non-interactive and information-theoretic secure verifiable secret sharing. In *Proceedings of the 11th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '91, pages 129–140, London, UK, UK. Springer-Verlag.

[Peter et al. 2013] Peter, A., Tews, E., and Katzenbeisser, S. (2013). Efficiently outsourcing multiparty computation under multiple keys. *IEEE Transactions on Information Forensics and Security*, 8(12):2046–2058.

[Peter et al. 2010] Peter, S., Westhoff, D., and Castelluccia, C. (2010). A survey on the encryption of convergecast traffic with in-network processing. *Dependable and Secure Computing, IEEE Transactions on*, 7(1):20–34.

[Reid and Harrigan 2013] Reid, F. and Harrigan, M. (2013). *An Analysis of Anonymity in the Bitcoin System*, pages 197–223. Springer New York, New York, NY.

[Santini 2005] Santini, S. (2005). We are sorry to inform you ... *Computer*, 38(12):128–127.

[Shor 1997] Shor, P. W. (1997). Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509.

[Vaccaro et al. 2007] Vaccaro, J. A., Spring, J., and Chefles, A. (2007). Quantum protocols for anonymous voting and surveying. *Phys. Rev. A*, 75:012333.

[Wang and Yu 2005] Wang, X. and Yu, H. (2005). How to break md5 and other hash functions. In *Proceedings of the 24th Annual International Conference on Theory and Applications of Cryptographic Techniques*, EUROCRYPT'05, pages 19–35, Berlin, Heidelberg. Springer-Verlag.

[Zheng and Huang 2013] Zheng, P. and Huang, J. (2013). An efficient image homomorphic encryption scheme with small ciphertext expansion. In *Proceedings of the 21st ACM International Conference on Multimedia*, MM '13, pages 803–812, New York, NY, USA. ACM.