Dispositivo quântico à prova de falsificação para transferência de prova

Alexandre Marques Albano da Silveira e José Cláudio do Nascimento

***Resumo*— Esse trabalho aplica o conceito de dispositivo quântico a prova de falsificação para realizar a prova de interação entre os dois agentes de um sistemas de prova de conhecimento nulo, o provador e o verificador. O ataque a impossibilidade de transferência de prova em sistemas de conhecimento nulo já é um resultado conhecido. Neste trabalho mostramos que para este feito é necessário apenas uma simples memória quântica. O ataque proposto neste trabalho não usa recursos de entrelaçamento como na primeira proposta.**

***Palavras-Chave*— Dispositivo quântico à prova de falsificação, prova de conhecimento nulo, isomorfismo de grafos, informação quântica.**

***Abstract*— This article applies the concept of quantum tamper-proof device to perform of the trace of interaction between two parts of a zero knowledge proof, the prover and the verifier. The attack to impossibility proof transferency in zero knowledges proof systems is already known result. In this work we show that is necessary only a simple quantum memory to perform this attack. The proposed attack does not use entanglement systems as the original proposal.**

***Keywords*— quantum tamper-proof device, zero knowledge proof, graph isomorfism, quantum information.**

I. INTRODUÇÃO

Stephen Wiesner escreveu o artigo intitulado *Conjugate Coding* em que fazia propostas inéditas de aplicação da mecânica quântica [1]. A idéia consistia basicamente em:

1) Produção de notas de dinheiro totalmente imunes à falsificação;
2) Um método para a combinação de duas mensagens em uma única transmissão quântica, de modo que o receptor pudesse escolher uma delas, mas não as duas simultaneamente. A leitura de uma implicava automaticamente na destruição da outra.

O artigo de Wiesner, publicado em 1983, lançou as bases da criptografia quântica. A idéia original de Wiesner para o dinheiro quântico era, em princípio teórico, possível, mas na prática era inviável, devido à dificuldade de armazenar fótons, manter os seus estados inalterados e não medidos por um longo período de tempo. No entanto, a idéia levantou um novo paradigma em sistemas de segurança. O dispositivo de Wiesner é essencialmente um dispositivo quântico à prova de falsificação (DQPF). Em essência, trata-se de uma máquina quântica em que os estados internos não podem ser acessados para impedir a previsão de resultados nas saídas.

Alexandre Marques Albano da Silveira e José Cláudio do Nascimento, Departamento de Engenharia Elétirca, Campus Sobral, Universidade Federal do Ceará, Sobral, Brasil, E-mails: claudio.nasce@gmail.com. Este trabalho foi parcialmente financiado pela CAPES, FUNCAP e CNPq.

Em [2], os autores destacam um DQPF para quebrar uma propriedade de sistemas de prova de conhecimento nulo. A idéia é construir um acordo entre duas partes que tira o poder de simulação de uma das partes. Assim, o anonimato do provador após uma interação com o verificador é quebrado. Watrous, em [3], destaca por que o paradigma da simulação não pode ser aplicado diretamente a verificadores com poder computacional quântico. As razões são as seguintes:

1) O estado quântico não pode ser copiado;
2) As medições são irreversíveis e seus efeitos não podem ser desfeitos.

Se o simulador é executado uma vez como uma caixa preta pelo verificador e a simulação não atinge um resultado satisfatório, não fica claro na simulação como reiniciar o processo e tentar novamente. Os estados de transição não podem ser copiados e a execução do verificador pode ter evoluído para um resultado irreversível.

Em [2], as propriedades desse dispositivo são discutidas e uma aplicação para assinaturas de contrato é apresentada. Neste trabalho, o DQPF é definido como um autômato quântico segundo a generalização de autômato quântico descrita em [4].

Definição 1: Dispositivo quântico à prova de falsificação Um dispositivo quântico à prova de falsificação é um autômato quântico formado pela tupla $Q = \{I, \Gamma, O, \mathcal{H}, |\psi_0\rangle, U, A\}$ em que:

1) I é um conjunto finito de símbolos de entrada;
2) Γ é um conjunto finito de símbolos de observação;
3) $O \subseteq \mathbb{R}$ é um conjunto finito de símbolos de saída;
4) $\mathcal{H}$ é um espaço de Hilbert de dimensão finita;
5) $|\psi_0\rangle \in \mathcal{H}$ é um vetor unitário em $\mathcal{H}$ chamado de estado inicial;
6) $U = \{U_i\}_{i \in I}$ é uma família de transformações unitárias em $\mathcal{H}$ chamada de família de transformações;
7) $A = \{A_\gamma\}_{\gamma \in \Gamma}$ é uma família de observáveis sobre $\mathcal{H}$, chamada de família de observáveis, tal que o espectro de A_γ está contido em O para todo $\gamma \in \Gamma$.

Estudando o poder computacional de um sistema físico podem-se extrair idéias significativas da estrutura e dinâmica do sistema. No autômato da definição 1, duas entradas são possíveis: entradas de transição $i \in I$ e entradas de observação $\gamma \in \Gamma$. Entradas de transição estão associadas às transformações $U = \{U_i\}_{i \in I}$, enquanto que as entradas de observação estão associadas aos observáveis, $A = \{A_\gamma\}_{\gamma \in \Gamma}$. Nesse caso, o espectro de A_γ está contido em O para todo $\gamma \in \Gamma$. Em outras palavras, o conjunto de saídas $O \subseteq \mathbb{R}$ contém os autovalores de todos os observáveis A_γ do autômato. Respei-

tando o postulado da mecânica quântica, o autômato inicia no estado $|\psi_0\rangle$, evolui deterministicamente quando as transições de entrada são lidas, e evolui probabilisticamente quando as entradas de observação são lidas. Assim, quando a transição i é lida, os estado $|\psi_0\rangle$ do autômato evolui para $U_i|\psi_0\rangle = |\psi_i\rangle$, sem apresentar qualquer valor na saída. Quando um símbolo de observação γ é lido, então o estado $|\psi_i\rangle$ evolui para $P_o|\psi_i\rangle/||P_o|\psi_i\rangle||$ com probabilidade $||P_o|\psi_i\rangle||$ para $o \in O$ de A_γ, em que P_o é a projeção do auto-espaço de A_γ associado à saída o.

II. Prova de conhecimento nulo

Agora será apresentado um pouco sobre isomorfismo de grafos para montar o cenário do próximo protocolo. Um grafo $G = (V, A)$ consiste de um conjunto de elementos chamados vértices $V = \{v_1, v_2, \cdots, v_n\}$ e um conjunto de pares de vértices chamados arestas, $(v_i, v_j) \in A$ para $i, j = 1, 2, \cdots, n$. Diz-se que um par de grafos $G_0 = (V_0, A_0)$ e $G_1 = (V_1, A_1)$ é isomorfo quando existe um mapeamento dos vértices do grafo G_0 para os vértices do grafo G_1, $\sigma : V_0 \rightarrow V_1$, tal que $(v_i, v_j) \in A_0$ se somente se $(\sigma(v_i), \sigma(v_j)) \in A_1$. Na prática, a função que realiza tal mapeamento de forma que o isomorfismo é sempre preservado é função de permutação dos vértices. Portanto, dado um grafo G com n vértices é possível gerar $n!$ grafos isomorfos a G, pois sempre existirá uma permutação inversa dos vértices que retorna para o grafo G, já que a permutação é uma função bijetiva. Encontrar um isomorfismo entre dois grafos, G_0 e G_1, é um problema difícil, mas fornecida a permutação dos vértices de G_0 que o torna igual a G_1, fica fácil verificar o isomorfismo entre estes grafos. O isomorfismo entre grafos é um problema que acredita-se não está em $\mathcal{P}$ [5]. Aqui essa classe de problemas será chamada $\mathcal{IG}$.

Para a compreensão do uso de isomorfismos de grafos como um desafio computacional, supõe-se que Paula diz a Victor conhecer um isomorfismo entre dois grafos isomorfos de n vértices, G_0 e G_1. Em se tratando de grafos de n vértices de uma forma geral, ele poderia tentar encontrar a solução na força bruta testando todos os grafos isomorfos a G_0, mas isso daria um algoritmo de complexidade em $O(n!)$ (embora algoritmos heurísticos melhorem esse resultado [6]). Portanto, conhecendo G_0 e G_1, Victor não tem como descobrir em tempo polinomial o isomorfismo, tal que $\sigma : G_1 \rightarrow G_0$. Com isso, Paula pode pegar um grafo G_1 (com n suficientemente grande) e um isomorfismo $\sigma : G_1 \rightarrow G_0$, para então divulgar G_0 e G_1 e ter como segredo o isomorfismo σ. Dessa maneira, será um segredo difícil de ser descoberto. Paula também sabendo que a composição de duas transformações isomorfas é um isomorfismo, pode escolher aleatoriamente vários isomorfismos para formar as composições e gerar vários grafos aleatoriamente. Agora, sendo Paula uma provadora ***P*** e Victor um verificador ***V*** que conhecem dois grafos G_0 e G_1 com n vértices, em que ***P*** tem como segredo o isomorfismo $\sigma : G_1 \rightarrow G_0$. O seguinte protocolo implementa um sistema interativo de prova de conhecimento nulo:

Protocolo 1: (Sistema de prova de conhecimento nulo para isomorfismos de grafos, [7]) - O seguintes passos são repetidos n vezes:

1) ***P*** gera um isomorfismo aleatório $\lambda : G_0 \rightarrow H$ e envia H para ***V***;
2) ***V*** gera aleatoriamente um bit b e envia o bit a ***P***;
3) ***P*** envia o isomorfismo $\xi = \sigma^b \circ \lambda$ para ***V***;
4) ***V*** confere se $\xi(G_b) = H$. (Se $b = 0$ implicará em $\lambda H = G_0$ e se $b = 1$ implicará em $\sigma \circ \lambda\ H = \sigma\ G_0 = G_1$).

Quando se olha o protocolo acima como um sistema de prova interativo, percebe-se que a entrada desse sistema é o par de grafos isomorfos G_0 e G_1. Então, a sentença dessa linguagem é $(G_0, G_1) \in \mathcal{IG}$. No caso do provador, o programa dele é executado por uma máquina probabilística de tempo polinomial, ou seja, apenas uma permutação escolhida aleatoriamente é realizada no Passo 1. O programa do verificador pode ser executado em tempo polinomial determinístico no passo 4. A escolha do bit de desafio é uma simples computação probabilística no Passo 2. Em [7] é mostrado que esse par de máquinas constitui um sistema de prova interativa com conhecimento nulo para a linguagem $\mathcal{IG}$ (Isomorfismo de Grafos).

Proposição 1: A linguagem $\mathcal{IG}$ tem um perfeito sistema de prova interativa com conhecimento nulo. Para verificadores limitados por máquinas de tempo polinomial (determinística ou probabilística), o Protocolo 1 satisfaz as seguintes afirmativas [5]:

1) Se G_0 e G_1 são isomorfos, então o verificador sempre aceita quando interage com ***P***;
2) Se G_0 e G_1 não são isomorfos, então a entrada será rejeitada com probabilidade menor do que $\frac{1}{2}$;
3) ***P*** realiza provas com perfeito conhecimento nulo.

Quando n interações entre o verificador e o provador são realizadas a probabilidade de erro para a validade é limitada por $1/2^n$. Deve ser enfatizado que todas as computações probabilísticas são completamente independentes para cada iteração das máquinas probabilísticas, conseqüentemente, isto é válido para interações. Outro fato a ser destacado é que não se sabe se a linguagem $\mathcal{IG} \in \mathcal{BPP}$ ou $\mathcal{IG} \notin \mathcal{BPP}$ [5]. Portanto, segue a prova da Proposição 1:

1) **Prova**: Claramente, se os grafos G_0 e G_1 são isomorfos, então o grafo H construído por ***P*** no Passo 1 é isomorfo a eles dois. Conseqüentemente se cada parte segue o que está prescrito no protocolo então ***V*** sempre aceita os argumentos do provador.
2) **Prova**: Se os grafos G_0 e G_1 não são isomorfos, então nenhum grafo pode ser isomorfo a G_0 e G_1. Isto segue que nenhum provador trapaceiro constrói H isomorfo ao dois simultaneamente, mas somente a um dos dois, ou seja, ele escolhe $b \in \{0, 1\}$ de forma que H vai ser isomorfo a um dos dois. Conseqüentemente, o verificador segue o programa e rejeita o argumento do provador com probabilidade 1/2.
3) **Prova**: Seja ***V*** um verificador eventualmente desonesto, isto é, um verificador que não segue necessariamente os passos do Protocolo 1. O objetivo é mostrar que é possível simular a interação deste verificador com um provador honesto, sem uma real comunicação com este provador. Primeiro observa-se que o objetivo do simulador é produzir n tuplas (H, b, ξ) onde H é gerado

uniformemente (pois o provador é honesto) e c é gerado de acordo com $\boldsymbol{V}$. Nota-se ainda que o simulador tem acesso ao código de $\boldsymbol{V}$. Nestas condições o verificador $\boldsymbol{V}$ é uma família de algoritmos probabilísticos em tempo polinomial $V_{k=1,2,\ldots,n}$ onde V_k corresponde ao algoritmos de escolha do bit de desafio na k-ésima iteração. O algoritmo V_k recebe o compromisso H e eventualmente poderá usar algum dado auxiliar $w_k \in \{0,1\}^*$ que calculou em iterações anteriores. No início $\boldsymbol{V}$ não tem nenhum dado auxiliar, ou seja, $w_k = \epsilon$ (sem símbolo). Então começa a simulação:

a) O simulador escolhe $b \in \{0,1\}$ de forma uniformemente aleatória;
b) O simulador gera o isomorfismo $\xi : G_b \rightarrow H$;
c) O simulador aplica $V_1(H, \epsilon)$ e verifica se o bit c calculado por V_1 é igual a b;

   i) Caso $c = b$, então a simulação foi feita com sucesso e a tupla nesta interação deve ser (H, c, ξ). A computação auxiliar feita por $\boldsymbol{V}$ com o fim de ser utilizada em interações futuras é gravada em w_1;
   ii) Se $c \neq b$, então o simulador volta para o Passo (a).

   Após a primeira iteração (k ¿1), no Passo (c), o simulador aplica $V_k(H, w_k)$ e verifica se o bit c calculado por V_k é igual a b;

   i) Caso $c = b$, então a simulação foi feita com sucesso e a tupla nesta iteração deve ser (H, c, ξ). A computação auxiliar feita por $\boldsymbol{V}$, com o fim de ser utilizada em iterações futuras, é gravada em w_{k+1};
   ii) Se $a \neq b$, então o simulador volta para o Passo (a).

Observe que a probabilidade de $c = b$ é $1/2$, dado que b foi escolhido com distribuição de probabilidade uniforme. Assim, o simulador terá sucesso, em média, após duas tentativas. Para finalizar, basta verificar que a seqüência de tuplas (H, c, ξ) gerada pelo simulador tem exatamente a mesma distribuição que as tuplas produzidas por uma interação real com o provador. Por esta razão, estas seqüências são computacionalmente indistinguíveis. □

Uma outra propriedade de sistemas interativos de conhecimento nulo é a *impossibilidade de transferência de prova*. Essa propriedade diz que após a interação entre o provador e o verificador, o anonimato do provador é preservado. Em outras palavras, um verificador com poder computacional de tempo polinomial não pode transferir uma prova da sua interação com o provador para uma terceira parte. O argumento para essa propriedade é bastante simples: Se o verificador consegue simular uma interação com o provador, então não conseguirá convencer com a sequência de tuplas $(H_i, c_i, \xi_i\}_{i=1}^n$ uma terceira, pois essa não saberá distinguir entre uma simulação e uma real interação a partir dessa seqüência.

III. Uma simples memória quântica transfere a prova de interação entre o provador e o verificador a uma terceira parte

A impossibilidade de transferência de prova é uma propriedade conhecida como resistente a ataques clássicos, mas foi mostrado em [8] que não é resistente a ataques quânticos com o uso de estados de Bell. Inicialmente, estados de Bell são preparados e a terceira parte, Eva, compartilha-os com o verificador. Durante o ataque, o verificador pode somente realizar dois tipos de medições nos qubits do par de Bell que estão em seu laboratório. Essa é a única entrada permitida no dispositivo quântico à prova de falsificação. Essas restrições são suficientes para impedir a simulação do verificador. Ao final, Eva tem como confirmar se o verificador foi honesto quanto ao uso do dispositivo.

Nesta seção, para entender a aplicação de uma memória quântica no impedimento de uma simulação, deve-se imaginar a seguinte situação: Eva armazena em uma memória quântica um estado $|\psi_a^b\rangle \in \{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$, em que b representa a base de medição (0 para a base $\{|0\rangle, |1\rangle\}$ e 1 para a base $\{|+\rangle, |-\rangle\}$) e a representa o valor lógico do qubit (0 se o estado quântico $|0\rangle$ ou $|+\rangle$ e 1 se o estado quântico é $|1\rangle$ ou $|-\rangle$). Os estados quânticos $|+\rangle$ e $|-\rangle$ são definidos pelas superposições: $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$ e $|-\rangle = (|0\rangle - |1\rangle)/\sqrt{2}$. Depois desse qubit estar armazenado em uma memória quântica, Eva envia a memória para Virgínia, que deve medir o estado e depois enviar o resultado da sua medição para Eva. Ressalta-se que Virgínia não conhece o estado quântico que está armazenado na memória quântica. Dessa maneira, Virgínia escolhe aleatoriamente uma base de medição β, realiza a medição, e obtém o resultado α. Realizada esta ação, Virgínia envia para Eva o par de valores (α, β). Ao receber estes valores, Eva confere se de fato a afirmativa condiz com a ação de Virgínia. Portanto, ela realiza a seguinte verificação:

1) Se $\beta \neq b$, então Eva descarta esse valor e assume que Virgínia está falando a verdade sem ter certeza disso;
2) Se $\beta = b$, então Eva verifica se $\alpha = a$. Caso realmente $\alpha = a$, então Eva tem a certeza de que Virgínia está falando a verdade. Caso contrário, Eva tem a certeza de que Virgínia está mentindo.

Nota-se que, nesse protocolo, o argumento de Virgínia é sempre aceito quando ela está falando a verdade. Por outro lado, quando Virgínia deseja mentir sobre a medição e o resultado que obteve, enviando o par $(\tilde{\alpha}, \tilde{\beta})$, ela sempre tem sucesso quando envia $\tilde{\beta} \neq \beta = b$, pois Eva não tem como verificar a sua afirmativa, então ela assume que Virgínia está falando a verdade. No entanto, quando Virgínia mente enviando $\tilde{\beta} = b \neq \beta$, então ela engana somente se ela envia $\tilde{\alpha} = a$. Mas se $b \neq \beta$, isso implica que o resultado da medição é completamente independente do valor a, ou seja, $\Pr(\alpha = a) = 1/2$, implicando que a escolha de $\tilde{\alpha}$ é uma escolha totalmente independente de a, ou seja, $\Pr(\tilde{\alpha} = a) = 1/2$. Portanto, quando Virgínia resolve mentir a respeito de sua medição, ela só conseguirá mentir com sucesso quando escolher $(\tilde{\alpha}, \tilde{\beta}) = (a, b)$, no que implica que

$$\Pr[(\tilde{\alpha}, \tilde{\beta}) = (a, b)] = \Pr[\tilde{\alpha} = a]\Pr[\tilde{\beta} = b] = 1/4. \quad (1)$$

Seja E o algoritmo de verificação de Eva e $\Pr(E(\alpha,\beta)_{aceita})$ a probabilidade de Eva aceitar o argumento de Virgínia, então esse simples algoritmo de verificação é um algoritmo probabilístico, satisfazendo as seguintes condições:

- Se Virgínia envia (α,β) para Eva, então $\Pr(E(\alpha,\beta)_{aceita}) = 1$;
- Se Virgínia envia $(\tilde{\alpha},\tilde{\beta}) \neq (\alpha,\beta)$ para Alice, então $\Pr(E(\tilde{\alpha},\tilde{\beta})_{aceita}) = \frac{3}{4}$.

Quanto ao fato de Eva aceitar entradas falsas, de forma mais precisa, percebe-se que

$$\begin{aligned}\Pr(E(\tilde{\alpha},\tilde{\beta})_{aceita}) &= \Pr[(\tilde{\alpha},\tilde{\beta}) = (a,b)] + \Pr[\tilde{\beta} \neq b] \\ &= \frac{1}{4} + \frac{1}{2} = \frac{3}{4}. \end{aligned} \quad (2)$$

Neste trabalho é feita uma simplificação em que somente qubits armazenados em memória quântica são usados para impedir a simulação do verificador. Será descrito o protocolo entre Eva, o verificador e o provador, que será usado para obter a evidência da interação entre o provador e o verificador em um sistema de prova de conhecimento nulo. O seguinte protocolo descreve um traço de interação entre o provador e o verificador que não pode ser falsificado, em que o provador conhece um isomorfismo entre os grafos G_0 e G_1 com n vértices.

Protocolo 2: (Sistema de prova interativa de conhecimento nulo com possibilidade de transferência de prova)

1) Eva e o verificador concordam com uma função hash $h : \mathcal{G}_n \to \{0,1\}^n$, em que $\mathcal{G}_n$ é o conjunto de todos os grafos isomorfos com n vértices.
2) Eva escolhe, com distribuição de probabilidade uniforme, as matrizes A e B escritas como:

$$A = \begin{pmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{pmatrix} \text{ e } B = \begin{pmatrix} b_{11} & \cdots & b_{1n} \\ \vdots & \ddots & \vdots \\ b_{n1} & \cdots & b_{nn} \end{pmatrix}. \quad (3)$$

Assim, Eva armazena os seguintes estados na memória quântica:

$$\begin{pmatrix} |\psi_{b_{11}}^{a_{11}}\rangle & \cdots & |\psi_{b_{1n}}^{a_{1n}}\rangle \\ \vdots & \ddots & \vdots \\ |\psi_{b_{n1}}^{a_{n1}}\rangle & \cdots & |\psi_{b_{nn}}^{a_{nn}}\rangle \end{pmatrix}. \quad (4)$$

Eva entrega a memória quântica com esses qubits armazenados para o verificador. Pode-se dizer, que essa máquina está selada para o verificador porque ele desconhece os estados internos da memória quântica. Agora, o verificador inicia a interação de prova de conhecimento nulo com o provador. Para cada interação com o provador, $k = 1,2,\ldots,n$, o protocolo é executado da seguinte maneira:

a) ***P*** gera um isomorfismo aleatório $\lambda_k : G_0 \to H_k$ e envia H_k para ***V***.
b) ***V*** recebe o grafo H_k enviado por ***P***, computa n bits usando a função hash, $h(H_k) = (h_{k1}, h_{k2}, \ldots, h_{kn})$, e realiza medições com a sequência obtida (h_{kl} é a base de medição do qubit $|\psi_{b_{kl}}^{a_{kl}}\rangle$) para obter a seqüência de observações na medição $o_k = (o_{k1}, o_{k2}, \ldots, o_{kn})$, com $o_{kl} \in \{0,1\}$ e $l = 1,2,\ldots,n$. Depois, ***V*** computa a paridade $p_k = \bigoplus_{l=1}^{n} o_{kl}$ e envia $p_k \in \{0,1\}$ para ***P***.
c) ***P*** recebe p_k enviado por ***V*** e envia o isomorfismo $\xi_k = \lambda_k \circ \sigma^{p_k}$;
d) ***V*** recebe o isomorfismo enviado por ***P*** e checa se $(G_(p_k)) = H_k$.

Quando a interação entre ***V*** e ***P*** finaliza, então ***V*** envia os resultados a partir do que foi medido da memória quântica junto com o traço de interação $\omega = (H_1, \xi_1, o_1)\ldots(H_n, \xi_n, o_n)$.

3) Para $k = 1,\ldots,n$, Eva checa se $\xi_k(G_{p_k}) = H_k$ e computa $h(H_k) = (h_{k1}, h_{k2}, \ldots, h_{kn})$. Para cada iteração, $l = 1,\ldots,n$, Eva fará como segue:

a) Caso $h_{kl} \oplus b_{kl} = 1$, Eva descarta b_{kl} e a_{kl} e assume que o verificador falou a verdade a respeito desta medição;
b) Caso $h_{kl} \oplus b_{kl} = 0$, então Eva checa se $o_{kl} = a_{kl}$. Se verdade, então Eva aceita o_{kl} e h_{kl} como prova e logo depois incrementa k, pois ela tem certeza de que o verificador falou a verdade a respeito desta medição. Caso contrário, ela rejeita a prova, pois ela tem certeza que o verificador está mentindo.

Se nunca houver rejeição no Passo 3, Eva obtém uma evidência da interação entre o verificador e o provador.

O Protocolo 2 traça a interação entre o verificador e o provador no sistema de prova interativa de conhecimento nulo com isomorfismo de grafo. Para mostrar que o ataque realmente quebra o anonimato do provador após a interação, vamos assumir que ***V*** deseja convencer Eva que ele realmente interagiu com ***P*** sem que isto tenha acontecido. Então, o objetivo do verificador é enviar a seqüência $\omega = (H_1, \xi_1, o_1)\ldots(H_n, \xi_n, o_n)$ para persuadir Eva de que a seqüência é um traço real de interação com o provador. Assim, com distribuição de probabilidade uniforme, o simulador escolhe aleatoriamente $c_k \in \{0,1\}$ e gera um isomorfismo aleatório $\xi_k(G_{c_k}) = H_k$. ***V*** calcula $h(H_k) \in \{0,1\}^n$ e usa seus bits para escolher as bases de medição, cujo resultados formam a seqüência de bits $o_k \in \{0,1\}^n$. Logo depois, ele calcula a paridade de $p_k = \oplus_{m=1}^{n} o_{km}$ e compara com c_k. Se eles são iguais, ***V*** envia (H_k, ξ_k, o_k) para Eva. Neste caso, que ocorre em metade dos grafos porque a probabilidade de $p_k = c_k$ é $1/2$, a simulação foi computada com sucesso. Por outro lado, para outra metade dos casos em que $p_k \neq c_k$, ***V*** pode pegar uma colisão de H'_k, tal que $h(H'_k) = h(H_k)$, calcula ξ'_k de forma que $\xi'_k(G_{c_k}) = H'_k$ e envia (H'_k, ξ'_k, o_k) para Eva. Apesar da probabilidade de encontrar colisões para função h ser negligenciável quando executado em tempo polinomial, pois o hash é uma função difícil de inverter, ainda assim será assumido que a imagem de h de cada grafo testado difere apenas de um bit do grafo original em testes em tempo eficiente.

O Passo 3 computado por Eva será denotado por $S_3(\cdot)$. De forma similar ao que foi descrito no protocolo em que Virgínia envia o resultado da medição de um estado quântico, quando o

verificador envia $(h(H'_k), o'_k) \neq (h(H_k), o_k)$ para Eva, então

$$\text{Prob}(S_3(h(H'_k), o'_k)_{aceita}) = 3/4. \quad (5)$$

Em outras palavras, a probabilidade de Eva aceitar a entrada $(h(H'_k), o'_k)$ como prova de interação é $3/4$, quando somente um bit é modificado na saída da função hash.

IV. Conclusões

Neste trabalho mostramos como uma simples memória quântica pode ser usada para atacar uma importante propriedade de sistemas de prova de conhecimento nulo. Neste ataque, o provador perde o anonimato após uma interação com o verificador em um algoritmo de prova de conhecimento nulo. O resultado aqui apresentado usa apenas processos de leitura e escrita em memória quântica para realizar tal ataque. A não-localidade, uma propriedade da mecânica quântica, não é usada no ataque aqui proposto. O objetivo é deixar claro que memórias quânticas são disposistivos a prova de falsificação e isto é suficiente para a transferência de prova do traço de interação entre o provador e o verificador.

Agradecimentos

A Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) e a Fundação Cearense de Apoio ao Desenvolvimento Científico e Tecnológico (FUNCAP).

Referências

[1] S. W., "Conjugate coding," *SIGACT News*, vol. 15, no. 1, pp. 78–88, 1983. [Online]. Available: http://dx.doi.org/10.1145/1008908.1008920

[2] J. Bouda, P. Mateus, N. Paunkovic, and J. Rasga., "On the power of quantum tamper-proof device," *International Journal of Quantum Information*, vol. 6, pp. 281 – 302, 2008.

[3] J. Watrous, "Zero-knowledge against quantum attacks," in *STOC '06: Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*. New York, NY, USA: Association for Computing Machinery, 2006, pp. 296–305.

[4] C. Moore and J. P. C. Eld, "Quantum automata and quantum grammars," *Theoretical Computer Science*, vol. 237, p. 2000, 2000.

[5] O. Goldreich, *Foundations of Cryptography: Basic Tools*. Cambridge University Press, 2001, vol. 1.

[6] J. Torán, A. T. Informatik, and O. Eselsberg, "On the hardness of graph isomorphism," in *SIAM J. Comput*. Society Press, 2000, pp. 180–186.

[7] O. Goldreich, S. Micali, and A. Wigderson, "Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems," *J. ACM*, vol. 38, no. 3, pp. 690–728, 1991.

[8] P. Mateus, F. Moura, and J. Rasga, "Transfering proofs of zero-knowledge systems with quantum correlations," *First International Conference on Quantum, Nano, and Micro Technologies*, vol. 0, p. 9, 2007.