# Minh ideas on Commitment scheme and Quantum Message Authentication Code from Graph States

Minh Thuy Truc Pham[1]
[1]*BTQ Technologies*

## CONTENTS

## I. COMMITMENT SCHEME FROM GRAPH STATE

### A. Commitment scheme

A commitment scheme is a cryptographic protocol that allows a party to commit the value while keeping it hidden from others with the chance to reveal it later. The Commitment scheme consists of two phases **commit** and **reveal** between the Commiter (Alice) and the verifier (Bob). a commitment scheme has two properties called "hiding" and "binding".

- **Hiding**: The commitment does not reveal any piece of information of the hidden value: *Unconditional hiding* and *Computational hiding.*

- **Binding**: Once a value has been committed to, the hidden value can not be changed: *Unconditional binding* and *Computational binding.*

  *Unconditional binding* means that with even infinite computing power, Alice can not change her mind (her information about the committed message) after sending the commitment.

  *Computational binding* means that unless Alice has "huge" computation resources, the chance of being able to change her mind is negligible.

A commitment scheme may have a similar form to Message Authentication but serves different purposes and has different properties. While both constructions are interactive, the message authenticator will send the message together with the tag to prove the authenticity of that message while the commitment scheme may hide the information of the actual message and just reveal it when needed.

### B. An example: Commitment scheme using Hash functions

Giving a cryptographic hash function $h$ that is preimage resistant and collision resistant. Alice wants to commit to a value $m$ with Bob. She first chooses a random string $r$ (nonce). This is to ensure the security of her commitment against brute force attacks. Alice then creates the commitment for $m$ by computing the hash of the concatenation of $m$ and $r$ as $c = h(m||r)$ but does not reveal these two values at this time. When needed, Alice reveals what she committed, and Alice sends the original values $m$ and $r$ to Bob. Bob performs the hash calculation $h(m||r)$ and then checks whether this value matches the committed value $c$ or not.

The security proof for the binding property of the above protocol is based on the **Rewinding technique**, a cryptographic tool used in the security proofs of many cryptographic primitives such as commitment schemes, interactive proofs, and zero-knowledge proofs. In this technique, a simulator completes a simulation or an attack by exploring different branches of the protocol execution.

**Rewinding** can be thought of as "reversing" the protocol to the previous stage (e.g., reversing the Turing tape in a Turing machine) and replaying it with different randomness or results.

The security proof proceeds through the following steps: Firstly, we assume that there exists an adversary $\mathcal{A}$ that can create two valid commitments $(m_1, r_1)$ and $(m_2, r_2)$ on the same commitment $c$. We can then use $\mathcal{A}$ to break the security of the hash function.

The simulator (verifier), after receiving the information $(c, m_1, m_2)$, saves this information and rewinds the protocol to the commitment step until $\mathcal{A}$ reveals $(m_2, r_2)$ satisfying $m_1 \neq m_2$, $r_1 \neq r_2$. The simulator now obtains two preimages of the hash function $h$, which violates the definition of $h$. Hence, the above scheme is (classically) secure.

### C. Quantum Commitment Scheme graph states

The quantum commitment scheme from graph states can be described as follows :

- **Commit**: Alice wants to commit a vector $\vec{p} \in \{0,1\}^{n(n-1)/2}$ . She encodes this vector into a graph $G = (V, E)$ of $n$ vertices where $\vec{p}$ is the vector of edges. Specifically, $p_i = 1$ if there is an edge between the corresponding vertices. Another way to think of this is that Alice commits to the adjacency matrix of a graph $G$. For this graph $G = (V, E)$, let $\rho(G)$ be the respective graph state. Alice now sends this graph state $\rho(G)$ to Bob as the commitment to the vector $\vec{p}$.

- **Reveal**: Alice reveals the structure graph $G$ (the set of stabilisers of $\rho(G)$). Bob then measures the state and yields $+1$ outcome if Alice is indeed committed to the vector $\vec{p}$.

**Security analysis**:

- **Hiding**: The quantum state $\rho(G)$ hide the information of the message vector unconditionally.
- **Binding**: Alice can not change the graph structure that she previously committed to.

## II. QUANTUM MESSAGE AUTHENTICATION CODE FROM GRAPH STATES

### A. Message Authentication Code

**Message Authentication Code (MAC)**, also referred to as "tag", is a piece of information sent along with a message from a specified sender to the receiver to ensure the **integrity** and **authenticity** of that message.

Formally, message authentication involves two parties, Alice and Bob, who share a secret key $k$ related to a specific function $f_k$. When Alice wants to send a message $m$ to Bob, she will compute the unique tag $t = f_k(m)$ associated with $m$ and send the pair $(m, t = f_k(m))$ to Bob. When Bob receives the message-tag pair $(m', t)$, he will recompute the authentication tag $f_k(m') = t'$ and check whether $t' = t$. If it matches, Alice's message $m$ is authenticated. The security requirement of the protocol is that adversaries who do not know about the secret key $k$ cannot create valid tags for messages they have never seen before.

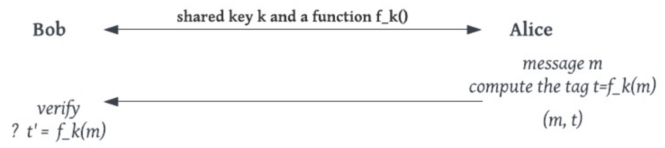

**Figure 1:** Message Authentication Code

### B. HMAC

**A Hash-based Message Authentication Code (HMAC)** is a type of Message Authentication Code (MAC) that employs a hash function to combine the message being authenticated with a secret key. This results in a unique hash value, the message tag, which can only be replicated if both the message and the key are known.

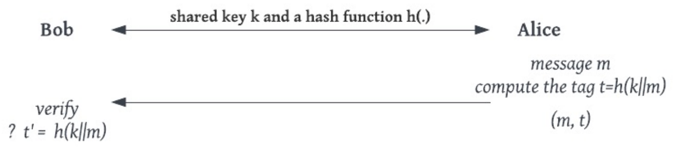
Bob
shared key k and a hash function h(.)
Alice
message m
compute the tag t=h(k||m)
verify
? t' = h(k||m)
(m, t)

**Figure 2:** Hash-based Message Authentication Code

It is important to note that in a message authentication key and tag are different to the key and signature in digital signature schemes due to the needed security properties.

- Integrity. The receiver is confident that the message has not been modified.
- Authenticity. The receiver is confident that the message originates from the sender.
- Non-repudiation property is a property that ensures that a party (sender or receiver) cannot deny having sent or received a message.
- Distinction between a digital signature and message authentication schemes:

Table I: Comparison between the HMAC and Digital signature

| | HMAC | Digital signature |
|---|---|---|
| Integrity | ✓ | ✓ |
| Authentication | ✓ | ✓ |
| Non-repudiation | ✗ | ✓ |
| Key | Symmetric | Asymmetric |

### C. Quantum Message Authentication Code(QMAC)

Let $\mathcal{X}_{2m+n}$ be the set of graphs with $2m + n$ vertices that satisfy:

1. The graph is separated into two subgraphs: one side with $m$ pairs of vertices (pair of messages vertices) and another side $G$ with $n$ vertices.

2. The subgraph $G$ has edges exists with probability $1/2$.
3. Each pair of message vertices $m_i$ is randomly connected to the vertices in the $G$ side such that it covers the whole space and randomly partitions it into two subgraphs.
4. The two vertices in each pair of message vertices are not connected.

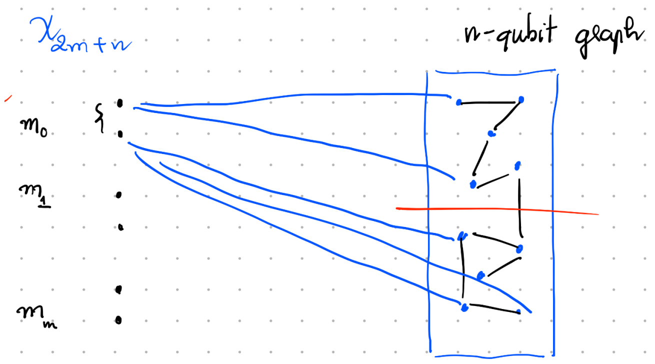

**Figure 3:** Graph structure of a graph $G' \in \mathcal{X}_{2m+n}$.

The quantum message authentication code scheme can be described as follows:

- **KeyGen**: Alice samples a graph state $G' = (V, E) \in \mathcal{X}_{2m+n}$ and send it to Bob. Note that both parties know the whole graph structure and the shared secret key is $(G', \rho(G'))$.
- **Authentication**: Alice wants to authenticate a message of $m$ bits.

  If $m_i = 0$, Alice measures the corresponding pair of qubits in the ZZ basis; otherwise, she measures them in the YY basis. The resulting graph is a $G$ of $n$ vertices. Alice now sends the graph state $\rho(G)$ as the tag of this message.

$$(m, t = \rho(G)) \tag{2.1}$$

- **Verification**: Receiving the message-tag pair from Alice, Bob now performs similar deletions and complementations of the graph structure according to the message on his secret key $G'$ to obtain a graph $G$. With the knowledge of all stabilisers of the graph $G$, Bob then measures the "tag" state $\rho(G)$. If this yields the $+1$ outcome, Alice's message is authenticated.

## REFERENCES