Noisy Byzantine Agreement Protocol in a Small Quantum Network

The Failure Probability of the Protocol Under Leakage Errors

Ayse Idil Evci[1]
Responsible Professor & Supervisor: Tim Coopmans[1]
[1]EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 22, 2025

Name of the student: Ayse Idil Evci
Final project course: CSE3000 Research Project
Thesis committee: Tim Coopmans, Arie van Deursen

Abstract

The Byzantine agreement problem is a challenge in distributed computing that explores how reliable parties can reach consensus even though there are unreliable or malicious participants. Quantum solutions propose better fault tolerance than the classical solutions, tolerating up to $t < \frac{n}{2}$ number of faulty or malicious parties, compared to the classical solutions with a limit $t < \frac{n}{3}$ where n is the number of participating parties. The Weak Broadcast Protocol we study proposes a quantum-aided solution using entangled four-qubit states. However, the hardware imperfections of quantum computers affect the reliability of the protocol. This paper investigates the impact of leakage errors, defined as unintended measurement outcomes that fall outside the set of expected basis states, on the failure probability of the protocol. We simulate the protocol in a noise-free setting using SquidASM to validate our setup and failure probabilities against the results reported in prior work. We then introduce a custom bit-flip leakage noise model, supported by an analytical formulation, and compare it with the pessimistic assumptions on the effect of the leakage errors by the prior work. Our results show that while the protocol's failure probability increases significantly under leakage noise, it is not as pessimistic as assumed before. The observed behavior differs for different fault configurations and shows that moderate noise may significantly affect the failure probability of the protocol. These findings suggest that the protocol is vulnerable to realistic noise.

1 Introduction

Quantum networks are expected to support applications such as secure communication and distributed quantum computing that can go further than what classical systems can do nowadays [12]. As these networks grow, it becomes important to ensure that nodes can reliably agree on shared values, although some components are malicious or faulty.

The Byzantine agreement problem is a specific problem in distributed decision-making that describes a situation where multiple parties must reach an agreement in presence of malicious or faulty parties, which are participants that does not behave as expected, either due to hardware imperfections or adversarial behavior [7]. Byzantine Agreement Protocols aim to solve this problem for n parties to reach a consensus. This problem is critical in practical systems such as secure communication, blockchain consensus, and coordination in quantum networks [12]. Even though classical Byzantine protocols are well studied, they cannot overcome the faulty behavior under certain bounds without increasing the communication complexity.

In classical systems, Byzantine agreement can tolerate up to $t < \frac{n}{3}$ faulty parties where n is the number of participating parties [10]. However, the quantum Weak Broadcast protocol studied in Guba et al. tolerate up to $t < \frac{n}{2}$ for correctness and security [6]. Quantum Byzantine protocol offers fault tolerance for fewer assumptions and quantum resource efficiency, as first explored by Fitzi [4]. The Weak Broadcast (WBC) protocol originally introduced by Fitzi and developed by Guba et al. is a quantum-aided solution using entanglement and quantum measurements to offer a stronger solution for the Byzantine agreement problem [4, 6].

Guba et al. focused primarily on analyzing the protocol's failure probabilities under various hardware imperfections, including measurement errors, gate noise, imperfect entanglement generation, and leakage errors [6]. However, their assumptions on leakage errors are treating each each leakage event equally harmful. In this work, we focus specifically on analyzing the effect of leakage errors under more realistic and selective assumptions by

simulating the protocol on a classical computer and isolating leakage from other types of hardware noise. Leakage errors, in the context of this paper, are a type of quantum noise where the system produces a measurement outcome in a basis state that lies outside the ideal set of basis states of the protocol. A four-qubit state can have 16 possible basis states, but only six are valid for this protocol. When the outcome corresponds to one of the ten invalid basis states, it is considered a leakage error.

To answer this, as a first step, WBC(3,1) protocol was implemented in SquidASM, a quantum network simulator, under noise-free conditions. The results were validated by comparing with the baseline findings of failure probability of the protocol from Guba et al. Then a custom leakage noise model was introduced and simulated. This paper analyzes how the leakage errors affect the protocol's failure probability across different faulty configurations.

Our main contribution with this research is a simulation of leakage errors and an analysis of how they affect the failure probability of WBC(3, 1). The leakage errors are simulated by a custom bit-flip noise model. The findings show that leakage errors increase the failure probability of the protocol significantly, however they are not as pessimistic as Guba et al. assumes [6]. With this research, we conclude that the protocol is not robust and does not perform well under leakage noise.

The remainder of this paper is structured as follows: Section 2 provides information about the background of WBC(3,1) and the simulation methodology. Section 3 outlines the problem statement including the definition of leakage errors, introduces the custom bit-flip noise model we implemented, and presents our analytical analysis of the noise model. Section 4 analyzes the experimental setup and presents our main findings under both noise-free and noisy settings for the different faulty configurations of the protocol. Section 6 outlines responsible research considerations. Section 5 analyzes and interprets the simulation results. Section 7 summarizes the conclusions and proposes directions for future work.

2 Background and Methodology

This section outlines the basics of quantum computing, implementation of WBC(3,1) protocol, explains the simulation approach used in SquidASM under noise-free conditions, and describes how faulty configurations were modeled. It also discusses how the failure probabilities were measured and the design of the leakage noise model.

The protocol WBC(3,1) has three parties involved including Sender S, Receiver R_0 and Receiver R_1. See Figure 2. It depends on a four-qubit entangled state, where sender performs measurements for two of the qubits and sends information to both receivers. If none of the parties are malicious or faulty, the expected behavior for the protocol to succeed is R_0 and R_1 agreeing on the information - bit-value, S sent. In case of having faulty or malicious parties it is acceptable that the protocol aborts and do not reach a consensus. The three configurations of this protocol - no-faulty, S-faulty, R_0-faulty - will be explained later in this chapter.

2.1 Quantum Computing

In this paper, we refer to basic information and notation from quantum computing concepts to be able to explain the methodology used to model and simulate the protocol.

An n-qubit quantum state can be represented by a complex vector of dimension 2^n. Each component of the quantum state corresponds to a basis state in the computational basis.

Quantum gates are represented as $2^n \times 2^n$ unitary matrices that act on the states (qubits) to perform transformations. Measurement in the computational basis of a qubit collapses its quantum state into a classical bit string. The qubits collapse to a certain state based on the probability amplitudes of the state vector which are coefficients of the basis states. These concepts are explained in detail in the textbook named "Quantum Computation and Quantum Information" by Nielsen and Chuang [9].

2.2 Qubit Teleportation

Quantum teleportation is a protocol that allows a qubit to be transmitted from one party to another without physically sending the qubit. This protocol is used in WBC(3,1) when the sender sends qubits to the receivers.

The protocol uses a shared two-qubit entangled state, specifically named Bell-pair, with some local operations and classical communication. First, an entangled pair is distributed between the sender and the receiver. Later, the sender entangles the qubit it wants to teleport, which is an unknown state $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, with their half of the Bell pair and performs a measurement. The outcome, which is two classical bits, is sent to the receiver, who applies Pauli corrections (I, X, Z, or XZ) to their half of the entangled pair. After these steps, the receiver's qubit is in the state $|\psi\rangle$, and the sender's original qubit has been destroyed by measurement. This process preserves the quantum state and shows that quantum information can be sent without physically relocating the qubit.

To see a more detailed explanation, the reader can check chapter 3.11 from the Qiskit Textbook by IBM [1].

2.3 Four-Qubit State

In the implementation of WBC(3,1) protocol, we used the four-qubit entangled state $|\psi\rangle$, originally proposed by Cabello [2].

$$|\psi\rangle = \frac{1}{2\sqrt{3}}(2|0011\rangle - |0101\rangle - |0110\rangle - |1010\rangle - |1001\rangle + 2|1100\rangle)$$

This state is used in the protocol because of the entanglement properties it has. It is non-separable, meaning it cannot be written as combination of different states. It also satisfies global unitary invariance, meaning the state remains unchanged when the same unitary operation is applied to all four qubits. This results in a symmetry that ensures measurement outcomes remain consistently correlated when measured in the same basis [2]. This entangled state was also used in the first experimental demonstration of a quantum protocol for Byzantine agreement and liar detection, conducted by Gaertner et al. [5].

Due to these properties, the state is the right choice for communication protocols using entanglement. We prepared the state using the Loop Circuit mentioned by Guba et al. [6].

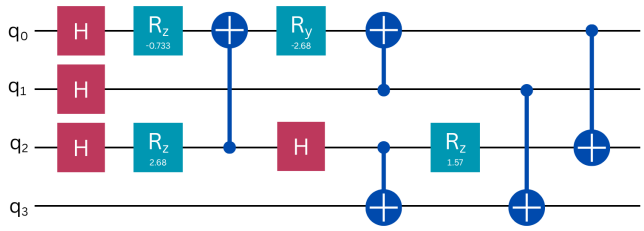

Figure 1: Preparation circuit for the four-qubit state used in WBC(3,1) based on the Loop Circuit from Guba et al. [6]

2.4 WBC(3,1) Protocol Description

In the protocol WBC(3, 1), the sender and the receivers share m number of four-qubit states, the specific state mentioned previously, and do their measurements m times. This is done by sender teleporting the two qubits to each of the receivers. This helps to statistically reduce the failure probability of the protocol and ensure that parties reach a consensus. The protocol is described in four stages by Guba et al. [6].

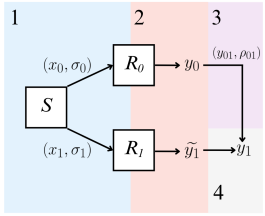

Figure 2: Illustration of the three parties involved in the WBC(3,1) protocol: Sender S, Receiver R_0, and Receiver R_1.

1. **Invocation Phase:** The sender S sends a classical bit $x_S \in \{0, 1\}$ to both receivers and creates m number of 4-qubit entangled states. It measures its two qubits from each state in the computational basis. If both measured bits equal to x_S, the corresponding index out of m is added to a *check set* σ_S, which is sent to both of the receivers. Receivers receive $x_j = x_S$ and $\sigma_j = \sigma_S$, and the sender sets its output to $y_S = x_S$.
2. **Check Phase:** Each receiver R_j measures the qubits indexed by σ_j out of m qubits they received. If all measurement results differ from x_j meeting the Consistency Condition, and $|\sigma_j| \geq T = \lceil \mu \cdot m \rceil$ meeting the Length Condition, the receiver accepts the message:
   - R_0 sets $y_0 = x_0$; otherwise $y_0 = \perp$ (abort).
   - R_1 sets intermediate value $\tilde{y}_1 = x_1$; otherwise $\tilde{y}_1 = \perp$.
3. **Cross-Calling Phase:** R_0 sends y_0 and σ_0 to R_1. R_1 receives them as y_{01} and ρ_{01}.
4. **Cross-Check Phase:** R_1 evaluates the received information with three conditions:
   (a) **Confusion Condition:** $y_{01} \neq \tilde{y}_1$, and $y_{01} \neq \perp$.
   (b) **Length Condition:** $|\rho_{01}| \geq T$.

(c) **Consistency Condition:** The number of indices in ρ_{01} where the measurement of R_1 differs from y_{01} is at least $\lambda T + |\rho_{01}| - T$.

If all conditions are satisfied, then $y_1 = y_{01}$; otherwise $y_1 = \tilde{y}_1$.

After completing the protocol, the goal is to ensure that the receivers reach a consensus on the value of the sender's bit. The success of the protocol is determined by the following conditions:

- **Validity:** If the sender S is honest and follows the protocol, then both honest receivers must agree on the original bit value x_S.
- **Consistency:** If any honest receiver outputs a bit $y \in \{0, 1\}$, then the other parties must on the same value y or abort. Disagreement among honest participants is not allowed unless some parties abort.

The four phases of the protocol work together to ensure secure and fault-tolerant agreement in the presence of a potential adversary. The invocation phase ensures that the receivers receive a check set from the sender so that they can verify the sender's bit by checking the consistency over the check set. The check phase allows receivers to individually verify the sender's message based on their own measurements. Without this step, a malicious sender could distribute inconsistent data. However, even if the check phase is present, disagreement between the receivers could still happen if the sender could manage to send incorrect information to one receiver. Cross-calling includes R_0 sending information to R_1, and the final cross-check phase enables them to resolve disagreements and reach a consensus. These phases are important, because the R_1 chooses to accept the value sent by R_0, or abort based on the consistency condition. These stages help to ensure that the validity and consistency conditions of the protocol are met.

The 4 phases mentioned above describe the no-faulty configuration of the protocol. For the no-faulty configuration, the protocol succeeds when $x_S = y_0 = y_1$. Two other configurations were implemented to model adversary strategies of the sender S and one of the receivers R_0 based on the descriptions by Guba et al [6].

- S-faulty: The sender acts maliciously by sending inconsistent information, that being $x_S = 0$ to one of the receivers and $x_S = 1$ to the other receiver. The protocol ensures the receivers can detect this malicious activity and either agree or abort. The failure condition is $y_0 \neq y_1$
- R_0-faulty: Receiver R_0 behaves maliciously, by providing false information to R_1. R_0 sends inconsistent bit value from x_S and a falsified check set during the Cross-Calling Phase. However, R_1 verifies this information using strict consistency and length conditions based on its own measurements. The protocol succeeds only when R_1 agrees with S.

2.5 Protocol Parameters μ and λ

The parameter μ determines the size of the check set which is used to validate the sender's consistency and λ sets the threshold for accepting a receiver's output during the cross-check phase.

Optimizing these values is important to minimize the failure probability of the protocol.

Guba et al. performed an empirical optimization over the (μ, λ) parameter space and figured that the values $\mu = 0.272$ and $\lambda = 0.94$ are an effective configuration that achieves a low failure probability considering all three configurations [6]. These values are within the theoretical region where the protocol is proven to be secure, allowing the failure probability to decay exponentially as the number of rounds increases. This is why we use the same values in our simulations.

2.6 SquidASM

In this research, SquidASM was used to implement and test the WBC(3,1) protocol under measurement noise.

SquidASM is a framework built higher-level on top of NetSquid that makes it easier to design and simulate distributed quantum protocols [11, 8]. NetSquid is a Python package that provides a simulator for quantum networks which uses discrete events to model how information is processed and shared between nodes [11, 8].

We implemented our simulation using SquidASM because it provides built-in features for setting up nodes, classical control, and adding noise. It also offers different device types which can be specified in the configuration. It offers a "generic" and a "Nitrogen-Vacancy (NV) Center" device. NV device has more fields to introduce noise, which will be discussed later in the Section 4.

3 Definition, Modeling, and Analytical Evaluation of Leakage Errors

In this section, we present the design and implementation of a custom leakage noise model for the WBC(3,1) protocol that was developed to isolate and study the effect of leakage errors on the failure probability of the protocol.

3.1 Problem Statement and Definition of Leakage Errors

In the context of this paper, leakage errors refer to having a measurement outcome not being from the intended set of "useful" states, but transforming into an unintended "useless" state. Specifically, in the context of the WBC(3,1) protocol, leakage occurs when the measurement of a four-qubit state yields a basis state that is not one of the six basis states with non-zero amplitude in the four-qubit state. The intended or "useful" states for the protocol are "0011, 0101, 0110, 1010, 1001, 1100".

In this research, we investigate how leakage errors, measurement outcomes that fall outside the set of six useful four-bit states from the ideal four-qubit state, impact the failure probability of the WBC(3,1) protocol. Addressing this question is important because useless measurement outcomes either violate the length condition or the consistency condition. Length condition is violated by having different measurement outcomes in the sender node, which then prevents the protocol from producing a large enough check set. The consistency condition is not met when a receiver has at least one measurement same as the sender among the indices specified in the check set.

By definition, this error can only be simulated by introducing small probabilities to measure a qubit incorrectly as $|0\rangle$ or $|1\rangle$, indicating that leakage errors represent measurement errors. Therefore, we address how "leakage" or "measurement" errors affect the failure probability of the protocol similarly.

3.2 Modeling of Leakage Errors

To be able to model the leakage errors, we considered two different approaches: Bit-flip Model and Kraus Operator Model. The Bit-flip Model changes the measurement outcome after the measurement process is done by small probability and the Kraus Operator Model applies noise directly to the quantum state before measurement, using matrices that capture the probabilities of different measurement outcomes due to imperfections. The reader can check Appendix C to learn more about the Kraus operators and Kraus Operator Model.

We assumed these two approaches would produce similar failure probability graphs for the protocol, because both noise models have the same probability of having a specific measurement outcome as mentioned in Table 1 in Appendix D [3].

After running the experiments with both noise models, we observe that the outcomes are unexpectedly different. The explanation of the Kraus Operator Model can be found in Appendix C with its mathematical formulation presented in Appendix C.1. The model is already implemented as a feature in SquidASM, and its logic is described in their paper [3]. Furthermore, we discuss our discoveries related to the difference between the Kraus Operator Model and Bit-flip model in Appendix D.

For the purpose of this research, we decided to further investigate the failure probability of the protocol simulating the bit-flip errors, because we wanted to have a full transparency and controllability over our noise model. Furthermore, in this model, we capture all possible measurement outcomes probabilistically, which makes it a simple approach, aligning with our definition of leakage, and allows for straightforward analytical analysis of its effect on the protocol.

3.3 Bit-flip Model

In the Bit-flip approach, to model the leakage errors, we represent them as independent bit-flip errors applied to each of the four qubits right after they are measured. This approach assumes that the quantum state itself is not corrupted before measurement.

In order to represent the probability of measuring a qubit to be state $|0\rangle$ or $|1\rangle$, we define p_0 and p_1 as follows:

- p_0: The probability of incorrectly measuring a qubit in state $|0\rangle$ as $|1\rangle$.
- p_1: The probability of incorrectly measuring a qubit in state $|1\rangle$ as $|0\rangle$.

The qubit is assumed to have been correctly collapsed into $|0\rangle$ or $|1\rangle$, and the classical outcome of the measurement is affected by p_0 or p_1 due to noise arising from the hardware imperfections in the readout process.

Applying these bit-flips to each qubit in a useful four-qubit basis state results in a bit string that is one of the unintended states formally defined as the leakage error with probability $q(p_0, p_1)$.

We derived an analytical expression of the bit-flip model for the no-faulty configuration to represent the protocol's failure probability under p_0 and p_1, and the derivation can be found in Section 3.4.

3.4 Bit-flip Model Analytical Failure Probability Under Noise

To model the failure probability of the WBC(3,1) protocol under asymmetric measurement noise, we analyze the effect of two different causes of the failure on the protocol: (1) the

check set being too small, failing the length condition, and (2) the presence of harmful leakage within a sufficiently large check set, failing the consistency condition.

We will base our calculations as $x_S = 0$, so the state is 0011, and the same calculations can be done for 1100. However, two scenarios are asymmetric due to different p_0 and p_1.

3.4.1 Probability of Measuring 0011 Under Noise

We first define the probability of measuring the basis state 0011 under bit-flip noise as:

$$p_{0011} = \frac{1}{3}\left[(1-p_0)^2(1-p_1)^2 + p_0^2 p_1^2 + (1-p_0)p_0(1-p_1)p_1\right]. \tag{1}$$

We consider that leakage can occur deviating from any one of the six useful states used in the four-qubit entangled state. This expression accounts for, correct measurement of 0011 without error, four-bit flip transformation of 1100 into 0011, two-bit-flip transformations of the other four useful states into 0011.

We cannot end up at the state 0011 from applying 1 or 3 bit-flips to any of the other useful states in the four-qubit state.

3.4.2 Failure Due to Check Set Size (l_T)

The required threshold check set size is defined as $T = \lceil \mu \cdot m \rceil$ by the protocol. The probability that fewer than T useful states out of m states were measured can be expressed as:

$$l_T^{0011} = \sum_{m_{0011}=0}^{T-1} \binom{m}{m_{0011}} p_{0011}^{m_{0011}} (1-p_{0011})^{m-m_{0011}}. \tag{2}$$

This is a binomial tail probability summing the chance that fewer than T rounds result in measurements to be included check set. This formulation is inspired by the analytical failure probability derivation for the no-faulty configuration presented by Guba et al. [6].

3.4.3 Failure Due To Inconsistencies in the Check Set

Even if the check set is sufficiently large, the protocol may still fail if any of the included indices are corrupted by a harmful leakage. When there is one inconsistent value the protocol fails to reach a consensus by definition. As $q(p_0, p_1)$ is the probability that a useful state leaks to a useless state, the probability that at least one harmful leakage occurs over m rounds is:

$$P_{\text{leak}}^{0011} = 1 - (1 - q^{0011}(p_0, p_1)))^m \tag{3}$$

We also analytically redefine the leakage probability $q(p_0, p_1)$ to include only the harmful events that can affect the protocol's success through corrupted check set entries.

We define the set of harmful measurement outcomes as the ones that can lead to inconsistency when the sender measures 00 and receivers' bits differ from the expected 11. The set of harmful measurements: $\{0010, 0001, 0000\}$.

When any of these outcomes occur, due to its own 00 measurement, the sender includes the corresponding index in the check set. However, the receivers do not observe a consistent 1 and 1 outcomes that contradict with the sender's bit. This leads to a failure in satisfying the consistency condition, thus contributes to the failure probability of the protocol.

We write the $q(p_0, p_1)$ as follows categorizing the leakage possibilities into different terms:

$$q^{0011}(p_0,p_1) = \underbrace{\frac{1}{3}\cdot(1-p_0)^2\cdot\left[1-(1-p_1)^2\right]}_{\text{Term 1}} \quad \underbrace{+\frac{1}{12}\cdot 4\cdot(1-p_0)\cdot p_1\cdot[1-p_0\cdot(1-p_1)]}_{\text{Term 2}} + \underbrace{\frac{1}{3}\cdot p_1^2\cdot\left(1-p_0^2\right)}_{\text{Term 3}}. \tag{4}$$

Each term represents probability of having a harmful measurement originating from intended states from the four-qubit state.

- Term 1: Original state 0011
- Term 2: Possible original states $\{0101, 0110, 1010, 1001\}$
- Term 3: Original state 1100

3.4.4 Combined Failure Probability

We now combine the two components representing the failure due to check set size and failure due to inconsistencies in the check set. When the the state is assumed to be 0011 he overall theoretical failure probability under asymmetric measurement noise is:

$$p_f^{0011} = l_T^{0011} + (1-l_T^{0011})\cdot\left[1-(1-q^{0011}(p_0,p_1)^m)\right], \tag{5}$$

This reflects the failure probability when the check set is too small (probability l_T) or it is large enough $(1-l_T)$ but it has inconsistent values due to any leakage event.

As mentioned before the same calculations can be done for the state 1100. To cover both scenarios where the selected state is 0011 either 1100, we can rewrite the analytical failure probability of the protocol as:

$$p_f^{(\text{noisy})} = \frac{1}{2}\cdot p_f^{0011} + \frac{1}{2}\cdot p_f^{1100} \tag{6}$$

4 Experimental Setup and Results

This section describes the experimental setup created to simulate the protocol under noise-free and noisy conditions for no, S-, and R0-faulty configurations and outlines the results of the simulations.

4.1 Experimental Setup

We conducted the simulations locally using the SquidASM framework and Python, parallelizing the simulations to multi-process Monte Carlo trials using available CPU cores [11].

We created a three-node network including three nodes representing a sender (S) and two receivers (R0 and R1), using NV center quantum device in our configuration. In the configuration file config.yaml, we disabled all hardware noise that can be specified, to isolate the effect of measurement leakage errors. Since we used the bit-flip model, the measurement errors were also set to zero.

We simulated leakage errors in the protocol by flipping measured bits with asymmetric probabilities: $p_0 = 0.05$ and $p_1 = 0.005$. The reason behind this selection is that NetSquid paper reported that the fidelities for state $|0\rangle$ is 0.95 and or state $|0\rangle$ is 0.995 [3]. This allowed us to have more direct control over the bit-flip model than what is available via SquidASM configuration field, which implements Kraus Operator Model.

We used $\mu = 0.272$ for the check set threshold and $\lambda = 0.94$ for the cross-check acceptance threshold as the protocol parameters as recommended by Guba et al. [6].

For each $m \in [20, 400]$, we executed $N = 100$ independent Monte Carlo trials and computed empirical failure probabilites based on the results.

Error bars in the plots from Figure 4 and Figure 3 represent the standard error of a Bernoulli distribution, computed by $\sqrt{p(1-p)/N}$, where p is the empirical failure probability observed across N trials for each value of m.

4.2 Reproducing the Protocol's Failure Probability Under Noise-Free Conditions

Before introducing any noise into the system, we validated the correctness of our implementation of the protocol by reproducing the noise-free results by Guba et al., specifically those presented in their Figure 4 [6]. They plot both the exact failure probabilities for the no-faulty configuration and the upper-bound failure probabilities for the S- and R0-faulty configurations of WBC(3,1). Their results are based on $N = 10{,}000$ Monte Carlo trials per value of m, which were compared against analytical estimates. Their Monte Carlo results are very close to the analytical probabilities.

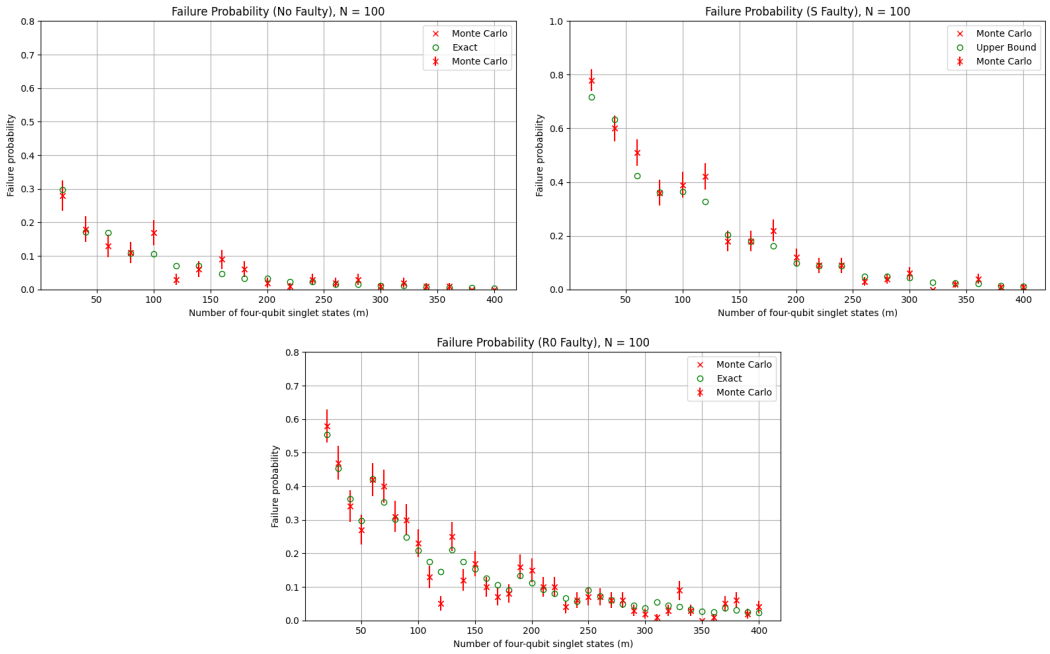

Figure 3: Simulation results under noise-free conditions: no-faulty, S-faulty, and R0-faulty configurations.

Due to computational limitations, we performed simulations with $N = 100$ trials per m, but our results closely follow the same trend as the exact and the upper bound curves reported in the paper by Guba et al. [6]. The exact and upper bound calculations are shown in the graphs in Figure 3 by the green circles.

The observed failure probability matches the expected trend as m increases. This confirms that our implementation correctly captures the baseline behavior of the protocol under noise-free conditions.

4.3 Failure Probability of the Protocol Under Bit-flip Modeled Leakage Errors

As for the purpose of this research, we investigated how the failure probability of the protocol is affected under the leakage noise.

In Figure 4, there are three graphs representing the failure probability of the protocol under leakage noise for different faulty configurations. In the first graph for the no-faulty configuration, the purple triangles mentioned as "Bit-flip Theoretical Model" are representing our analytical derivation of the failure probability under leakage noise. The "Noisy Model" mentioned on the graphs and plotted by blue squares represent Guba et al.'s pessimistic assumption of leakage errors [6].

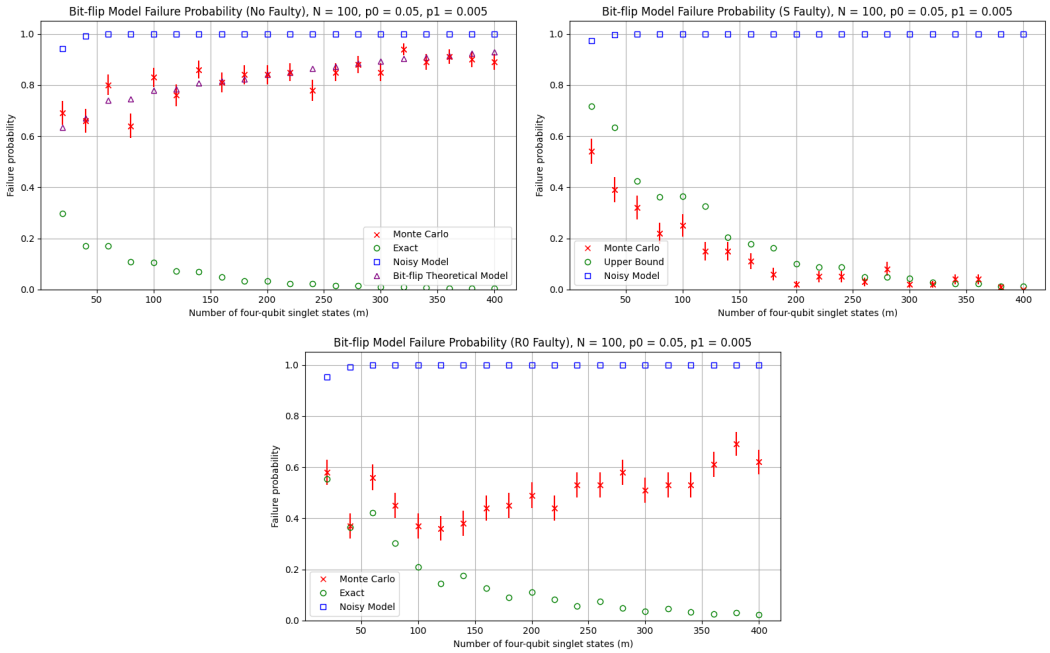

Figure 4: Simulation results under bit-flip model: no-faulty, S-faulty, and R0-faulty configurations.

For the no-faulty configuration, we observe that the bit-flip model produces less pessimistic results. The failure probability still increases as m increase, however at a slower rate. The pessimistic model by Guba et al. approaches to not reaching to a 100% failure

probability exponentially. The trend observed with the bit-flip model appears to be closer to a linear trend.

According to our observations, the S-faulty configuration does not show increase in failure probability under bit-flip noise. The failure probability remains below the upper bound values derived for the noise-free scenario, suggesting that the protocol is less sensitive to leakage errors in the S-faulty configuration.

The R0-faulty configuration shows increase in the failure probability of the protocol under leakage noise. However, the increase is not as much as the no-faulty configuration. The failure probability increases with a linear-looking trend.

5 Discussion

This section analyzes the simulation results and trends in failure probability under leakage noise and compares them with the noise-free baseline.

5.1 Reliability of the Monte Carlo Simulation Results

Although our simulations used a smaller number of Monte Carlo trials compared to Guba et al., respectively $N = 100$ and $N = 10,000$, the trends that were produced for the noise-free baseline closely match the exact and upper-bound failure probabilities. Furthermore, the error bars we use are representing the standard error of mean as mentioned in Section 4.

Across the values of m, the error bars are narrow relative to the range of the y-axis and do not overlap across significantly different values of m. This shows that the observed trends are statistically reliable. As a result, we concluded that running the simulation with $N = 10,000$ was not necessary and proceeded with $N = 100$.

5.2 No-Faulty Configuration Under Bit-Flip Noise

According to our findings, the no-faulty configuration shows a significant increase in the failure probability as the number of rounds m increases. This is because, in the no-faulty configuration, the protocol only succeeds if all three parties accept the same value. Even a single leakage event can introduce an index in the check set that either does not correctly reflect the sender's measurements or fails to meet the consistency requirements for the receivers, leading to failure.

When we compare our results to the pessimistic assumption by Guba et al., the reason we observe a less pessimistic failure probability is that we redefine leakage probability specifically based on events that may actually corrupt the protocol, rather than treating any leakage as equally harmful. Additionally, we distinguish between two failure cases: one due to an insufficient check set size (length condition) and one due to inconsistencies in the check set (consistency condition). Guba et al. do not separate these cases and instead assume that any leakage event can cause the protocol to fail [6].

5.3 S-Faulty Configuration Under Bit-Flip Noise

In the S-faulty configuration, where the sender is sending inconsistent bits and check sets to the two receivers, a failure is only recorded if both receivers accept and output different measurement values. The success and failure cases of the protocol can be observed in the Figure 8 in Appendix A. Under bit-flip noise, our results show a slight decrease in the number

of such failures. This appears as a counterintuitive outcome. However, it arises from the fact that leakage noise increases the chance that at least one receiver fails the consistency check or receives a check set that is too small, leading them to abort. Since aborting is a successful outcome in the S-faulty configuration, originating from the idea that deception was not successful, the presence of leakage noise actually contributes to the success cases in this configuration.

To summarize, these result show that in the S-faulty configuration, moderate noise can reduce the failure probability by increasing the number of events that at least one receiver aborts, preventing the adversary from successfully deceiving both parties.

5.4 R0-Faulty Configuration Under Bit-Flip Noise

In the R0-faulty configuration of the protocol, the protocol only succeeds when R_1 does not abort and accept the same value as the sender. Any other outcome is considered as a failure. Our observations show that R0-faulty case is also affected significantly by the bit-flip noise applied. Bit-flips that can occur in the sender's measurements have a similar effect as the no-faulty case, which may either make the size of the check set too small to satisfy the length condition or introduce indices that later cause consistency failures. When the bit-flip occurs in R_1's measurements, this can contribute to R_0's adversary strategy, because R_1 will be more likely to be deceived and accept the incorrect value sent by R_0 or abort.

Another point we must consider is that due to our implementation always setting the sender's bit to $x_S = 0$, the failure probabilities are slightly lower than randomizing x_S or setting $x_S = 1$. We discuss further in Appendix B that, due to the asymmetry in measurement fidelities for states $|0\rangle$ and $|1\rangle$, measurement errors on the state 0011 are less likely to violate consistency condition than those on 1100.

In conclusion, R0-faulty case is also significantly affected by the leakage errors and the failure probability of the protocol increases over increasing m.

6 Responsible Research

This section discusses reproducibility, ethical considerations, and our use of Large Language Models (LLMs).

In order to have reproducible research, it is important to be transparent with the methodology and results. We explained the methodology and experimental setup including all the parameters, chosen values and the mathematical formulations. Since this research is based on simulations, the results presented are inherently stochastic. With larger sample size, more accurate estimates can be produced.

For the base case scenario, simulation results are compared with analytical values to verify that they follow the expected trends. However, there are no analytical comparisons for the noisy simulations. Although analytical expressions are not provided and the number of simulation runs are relatively small, the results follow expected theoretical trends. Each configuration was simulated over 100 runs with error bars included to represent statistical uncertainty. The error bars are calculated using the Binomial Standard Error, which accounts for the variance in repeated independent Bernoulli trials.

To further support reproducibility, the simulation code is available the following GitHub repository: https://github.com/ayseidilevci/noisy_byzantine_agreement.git.

The research does not involve any data privacy concerns since it depends on the simulation data results rather than human subjects or personal data.

During the research, we used LLMs such as ChatGPT to improve clarity, structure, and grammar. Our technical content, methodology or analysis are the original work of the authors.

7 Conclusions and Future Work

This section summarizes key findings regarding the impact of leakage errors on the WBC(3,1) protocol and provides an overview of the next steps for future research and improvement.

7.1 Conclusion

In this work, we analyzed the effect of leakage errors on the failure probability of the WBC(3,1) protocol. We proposed a custom bit-flip noise model that allows us to have control over asymmetric measurement error probabilities and offers a transparent and analyzable methodology for simulating leakage. We also derived an analytical expression for the failure probability of the protocol under this model.

We reproduced the noise-free failure probabilities originally presented by the Figure 4 from Guba et al. and took it as a baseline, confirming the correctness of our implementation [6]. Although we used a smaller number of Monte Carlo trials ($N = 100$), we showed that the trends remain statistically reliable as confirmed by narrow error bars. After validating our experimental setup we went deeper in the analysis of leakage errors.

Under the bit-flip noise model, the no-faulty configuration had a steady increasing trend in failure probability with increasing m, consistent with our theoretical expectations. However, by refining the definition of harmful leakage and separating failure causes into length and consistency violations our model resulted in less pessimistic results than the assumptions made by Guba et al [6].

In the S-faulty configuration, we observed a slight decrease in failure probability under noise. This effect is because the likelihood that at least one receiver aborts due to inconsistencies or insufficient check set size increases and aborting is treated as a success scenario originating from the idea that faulty sender could not deceive the receivers to agree on different values.

The R0-faulty configuration also showed a significant increase in the failure probability due to leakage. In this case, bit-flips in either the sender's or R_1's measurements can make it easier for the adversarial strategy of R_0 to reach its goals or disrupt honest consensus.

In summary, our findings show that the leakage errors degrade the protocol's performance significantly and the protocol is not very robust and resilient against noise.

7.2 Future Work

An unanswered question that remains from our research is why the bit-flip model produces different results from the Kraus Operator model implemented by SquidASM, even though they have the same probabilities for producing the same basis state outcomes. A deeper investigation into the implementation details of SquidASM, specifically the NV center device configuration and how the noise is applied, and a formal mathematical comparison between the two noise models would be an important direction for future research.

Additionally, according to our analysis the failure probability of the WBC(3, 1) protocol increases significantly under the leakage noise and this shows us that the protocol lacks

robustness when exposed to hardware noise. Exploring alternative ways of designing the protocol that is more resilient to errors would be a significant improvement.

References

[1] Amira Abbas, Stina Andersson, Abraham Asfaw, Antonio Corcoles, Luciano Bello, Yael Ben-Haim, Mehdi Bozzo-Rey, Sergey Bravyi, Nicholas Bronn, Lauren Capelluto, Almudena Carrera Vazquez, Jack Ceroni, Richard Chen, Albert Frisch, Jay Gambetta, Shelly Garion, Leron Gil, Salvador De La Puente Gonzalez, Francis Harkins, Takashi Imamichi, Pavan Jayasinha, Hwajung Kang, Amir H. Karamlou, Robert Loredo, David McKay, Alberto Maldonado, Antonio Macaluso, Antonio Mezzacapo, Zlatko Minev, Ramis Movassagh, Giacomo Nannicini, Paul Nation, Anna Phan, Marco Pistoia, Arthur Rattew, Joachim Schaefer, Javad Shabani, John Smolin, John Stenger, Kristan Temme, Madeleine Tod, Ellinor Wanzambi, Stephen Wood, and James Wootton. Learn quantum computation using qiskit, 2020.

[2] Adán Cabello. Solving the liar detection problem using the four-qubit singlet state. *Phys. Rev. A*, 68:012304, Jul 2003.

[3] Tim Coopmans, Robert Knegjens, Axel Dahlberg, David Maier, Loek Nijsten, Julio de Oliveira Filho, Martijn Papendrecht, Julian Rabbie, Filip RozpÄdek, Matthew Skrzypczyk, Leon Wubben, Walter de Jong, Damian Podareanu, Ariana Torres-Knoop, David Elkouss, and Stephanie Wehner. Netsquid, a network simulator for quantum information using discrete events. *Communications Physics*, 4(1):164, 2021.

[4] Matthias Fitzi. *Generalized Communication and Security Models in Byzantine Agreement*. PhD thesis, ETH Zurich, 2003. Reprint as vol. 4 of ETH Series in Information Security and Cryptography, Hartung-Gorre Verlag.

[5] Sascha Gaertner, Mohamed Bourennane, Christian Kurtsiefer, Adán Cabello, and Harald Weinfurter. Experimental demonstration of a quantum protocol for byzantine agreement and liar detection. *Phys. Rev. Lett.*, 100:070504, Feb 2008.

[6] Zoltán Guba, István Finta, Ákos Budai, Lőránt Farkas, Zoltán Zimborás, and András Pályi. Resource analysis for quantum-aided byzantine agreement with the four-qubit singlet state. *Quantum*, 8:1324, April 2024.

[7] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 4(3):382–401, 1982.

[8] NetSquid Forum Community. Netsquid forum. https://forum.netsquid.org/index.php?sid=d6838f9776b58c6212fe22b1bd6bdc4a, 2025. Accessed: June 2025.

[9] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information*. Cambridge University Press, 10th anniversary edition edition, 2002.

[10] Marshall Pease, Robert Shostak, and Leslie Lamport. Reaching agreement in the presence of faults. *Journal of the ACM (JACM)*, 27(2):228–234, 1980.

[11] QuTech. SquidASM: A quantum network protocol simulator built on NetSquid. https://github.com/QuTech-Delft/squidasm, 2024. Accessed: June 2025.

[12] Stephanie Wehner, David Elkouss, and Ronald Hanson. Quantum internet: A vision for the road ahead. *Science*, 362(6412):eaam9288, 2018.

A Protocol Success Condition Truth Table for Different Faulty Configurations

S	R_0	R_1	no faulty	S faulty	R_0 faulty
0	0	0	✓	✓	✓
0	0	1	×	×	×
0	0	⊥	×	✓	×
0	1	0	×	×	✓
0	1	1	×	✓	×
0	1	⊥	×	✓	×
0	⊥	0	×	✓	✓
0	⊥	1	×	✓	×
0	⊥	⊥	×	✓	×
1	0	0	×	✓	×
1	0	1	×	×	✓
1	0	⊥	×	✓	×
1	1	0	×	×	×
1	1	1	✓	✓	✓
1	1	⊥	×	✓	×
1	⊥	0	×	✓	×
1	⊥	1	×	✓	✓
1	⊥	⊥	×	✓	×

Figure 5: Truth table for the success conditions of the protocol from Guba et al. [6].

B Effect of Sender Bit Choice on Failure Probability

Our implementation of the protocol for the no-faulty configuration randomizes the bit value of $x_S = 0$ and $x_S = 1$, and for S-faulty configuration it purposefully sends both 0 and 1 to different receivers. However, in the R0-faulty case, the sender always picks $x_S = 0$ and R_0 creates an adversarial check set assuming itself and R_1 should measure 1. For the noise-free cases, since the protocol is symmetric, this does not cause any problems, however, under asymmetric bit-flip noise, the failure probability of the protocol is affected based on the value chosen for x_S.

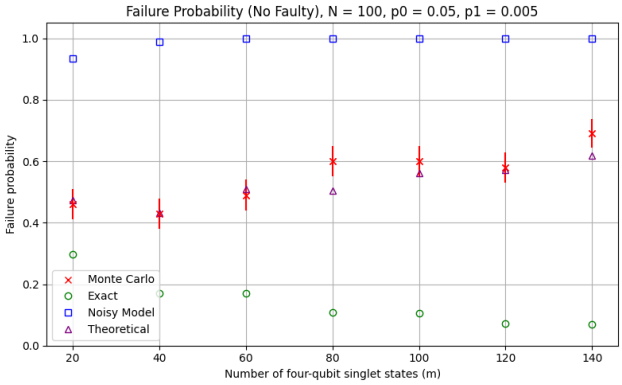

Figure 6: $x_S = 0$ No-Faulty Configuration Under Bit-flip Noise Model.

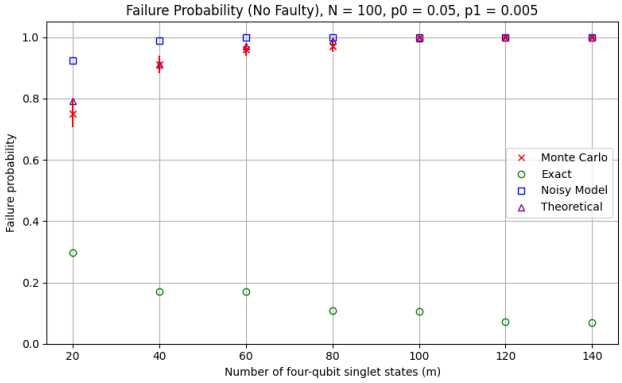

Figure 7: $x_S = 1$ No-Faulty Configuration Under Bit-flip Noise Model.

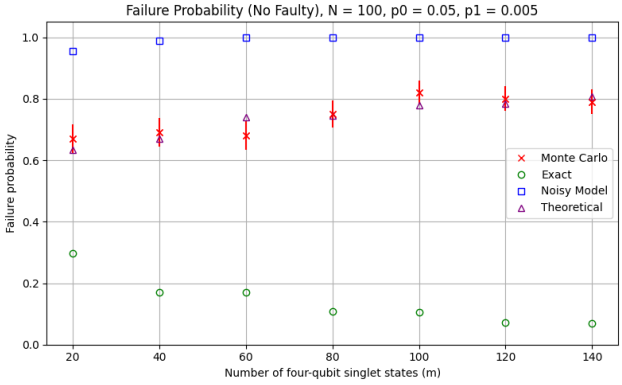

Figure 8: $x_S = 0, 1$ No-Faulty Configuration Under Bit-flip Noise Model.

According to our experimental observations and mathematical derivations, the protocol simulated with the preferred state 0011 to create the check set has significantly lower failure probability compared to the preferred state 1100. The reason behind is that, the failure probability of the protocol depends on both the length condition and consistency condition, but mathematically it is mostly dominated by the consistency condition. Since the fidelity of $|0\rangle$ is less than the fidelity of $|1\rangle$, setting $x_S = 0$ increases the failure probability by length condition, but receivers' having to measure their qubits as 1 decreases the failure probability due to consistency condition. Since consistency condition dominates length condition, selected measurement outcome $|0011\rangle$ to create a check set leads to lower failure probability than $|1100\rangle$.

C Kraus Operator Model

The Kraus operator model simulates leakage by applying a quantum noise process that probabilistically flips the qubit state before measurement, modifying the state itself rather than only the classical outcome. This approach simulates leakage errors using the stack type Nitrogen-Vacancy (NV) center qdevice in SquidASM which applies noise through the Kraus Operators. It represents the decoherence processes that occur prior to measurement.

This model assumes that noise occurs before the measurement, at the quantum level, thus the qubit's state may be disrupted before the measurement, leading to incorrect measurement outcomes even with a perfect measurement device.

The noise process can be described using two Kraus operators M_0 and M_1, which flip the state from $|0\rangle$ to $|1\rangle$ and vice versa.

$$M_0 = \begin{pmatrix} \sqrt{f_0} & 0 \\ 0 & \sqrt{1-f_1} \end{pmatrix}, \quad M_1 = \begin{pmatrix} \sqrt{1-f_0} & 0 \\ 0 & \sqrt{f_1} \end{pmatrix}$$

These operators satisfy the completeness relation $M_0^\dagger M_0 + M_1^\dagger M_1 = I$, making it a valid quantum channel. The parameters f_0 and f_1 represent the fidelities of correctly preserving the $|0\rangle$ and $|1\rangle$ states.

This approach follows the error modeling used in the NV-center configuration of NetSquid [3]. A detailed mathematical derivation of this model is provided in Appendix C.1. The probability of measuring a qubit as $|0\rangle$ and $|1\rangle$ are shown as follows:

$$p_0 = f_0|\alpha|^2 + (1-f_1)|\beta|^2$$
$$p_1 = (1-f_0)|\alpha|^2 + f_1|\beta|^2$$

The probabilities above result from applying the Kraus operators to any quantum state $|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$ and computing the diagonal entries of the resulting density matrix.

The values f_0 and f_1 are fidelity values of measuring the qubits correctly. Based on experimentally reported readout fidelities for NV center quantum devices, for the state $|0\rangle$ the fidelity value is 0.95 and for the state $|1\rangle$ it is 0.995 [3].

C.1 Derivation of Measurement Probabilities Using Kraus Operators

We consider a qubit state represented as:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle, \quad \text{where } |\alpha|^2 + |\beta|^2 = 1$$

The density matrix of this pure state is:

$$\rho = |\psi\rangle\langle\psi| = \begin{pmatrix} |\alpha|^2 & \alpha\beta^* \\ \alpha^*\beta & |\beta|^2 \end{pmatrix}$$

To model a noisy measurement in the computational basis with potential measurement errors, we use two Kraus operators:

$$M_0 = \begin{pmatrix} \sqrt{f_0} & 0 \\ 0 & \sqrt{1-f_1} \end{pmatrix}, \quad M_1 = \begin{pmatrix} \sqrt{1-f_0} & 0 \\ 0 & \sqrt{f_1} \end{pmatrix}$$

These operators correspond to a POVM measurement, and they satisfy:

$$M_0^\dagger M_0 + M_1^\dagger M_1 = I$$

C.2 Probability of outcome 0

The probability of measuring outcome 0 is:

$$p_0 = \mathrm{Tr}(M_0^\dagger M_0 \rho)$$

First compute:

$$M_0^\dagger M_0 = \begin{pmatrix} f_0 & 0 \\ 0 & 1-f_1 \end{pmatrix}$$

Then compute the matrix product:

$$M_0^\dagger M_0 \rho = \begin{pmatrix} f_0 & 0 \\ 0 & 1-f_1 \end{pmatrix} \begin{pmatrix} |\alpha|^2 & \alpha\beta^* \\ \alpha^*\beta & |\beta|^2 \end{pmatrix} = \begin{pmatrix} f_0|\alpha|^2 & f_0\alpha\beta^* \\ (1-f_1)\alpha^*\beta & (1-f_1)|\beta|^2 \end{pmatrix}$$

Now take the trace:

$$p_0 = \mathrm{Tr}(M_0^\dagger M_0 \rho) = f_0|\alpha|^2 + (1-f_1)|\beta|^2$$

C.3 Probability of outcome 1

Similarly, for outcome 1:

$$p_1 = \mathrm{Tr}(M_1^\dagger M_1 \rho)$$

with

$$M_1^\dagger M_1 = \begin{pmatrix} 1-f_0 & 0 \\ 0 & f_1 \end{pmatrix}$$

$$M_1^\dagger M_1 \rho = \begin{pmatrix} (1-f_0)|\alpha|^2 & (1-f_0)\alpha\beta^* \\ f_1\alpha^*\beta & f_1|\beta|^2 \end{pmatrix}$$

Taking the trace:

$$p_1 = (1-f_0)|\alpha|^2 + f_1|\beta|^2$$

C.4 Final Result

$$\boxed{\begin{aligned} p_0 &= f_0|\alpha|^2 + (1-f_1)|\beta|^2 \\ p_1 &= (1-f_0)|\alpha|^2 + f_1|\beta|^2 \end{aligned}}$$

C.5 Kraus Operator Noise Model Simulation Results

The same implementation of the WBC(3,1) protocol used in the noise-free evaluation was used under the Kraus operator noise model introduced in Section 4.

To simulate the quantum measurement errors, we used the SquidASM simulator with its built-in support for modeling device level noise through Kraus operators. In particular, we configured the simulation to use the NV quantum device model by specifying the device type and error parameters in the YAML configuration file used by SquidASM.

```
prob_error_0: 0.05
prob_error_1: 0.005
```

These parameters correspond to the probability of bit-flip errors occurring for qubits initially in states $|0\rangle$ and $|1\rangle$. They reflect experimentally reported readout error rates based on the fidelity values for NV-center-based quantum devices which is taken from the NetSquid paper [3].

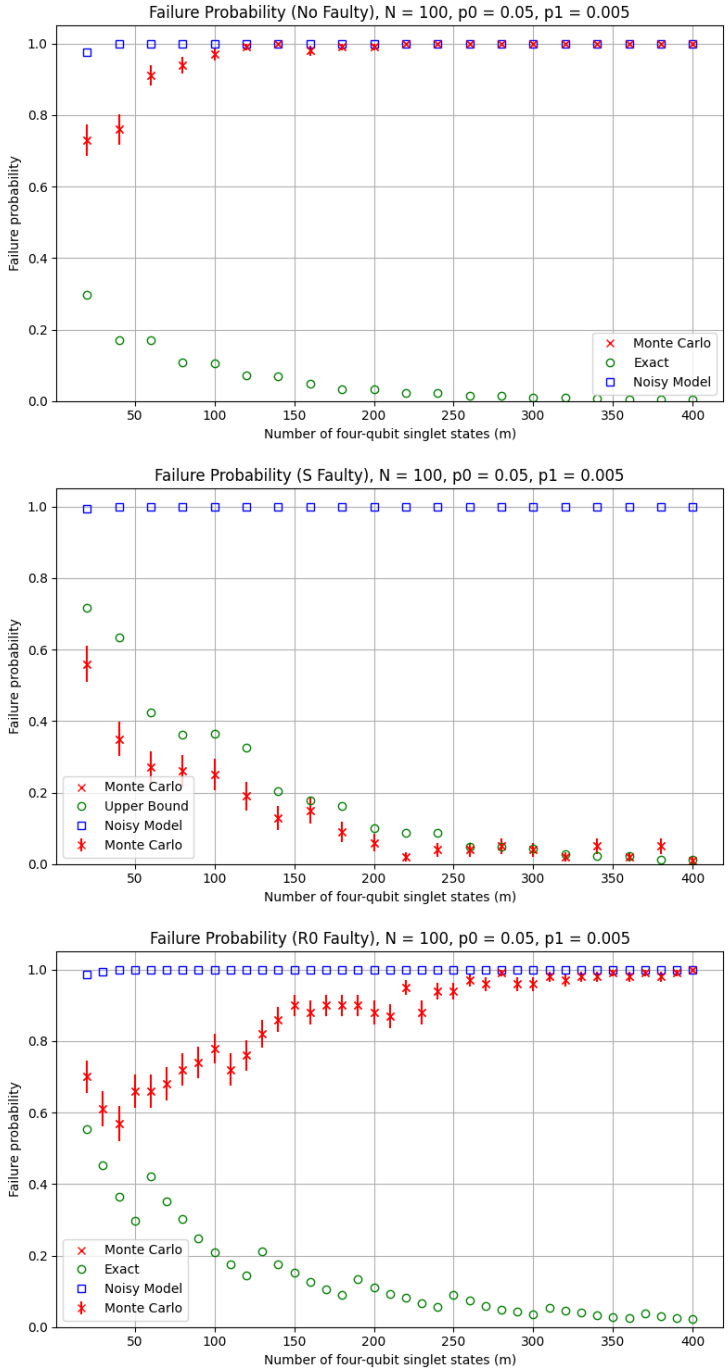

Figure 9: Failure probability of the protocol under leakage errors implemented by Kraus Operator Model.

Figure 9 shows the Monte Carlo simulation results for the failure probability of the

WBC(3,1) protocol under the Kraus operator noise model, with the theoretical predictions from Guba et al, which are shown as the blue squares called the "Noisy Model". [6].

In the no-faulty configuration, the failure probability increases significantly due to the noise, closely aligning with the pessimistic theoretical upper bound proposed in [6], especially after$m = 100$.

For the R_0-faulty configuration, a less drastic increase in failure probability is observed. However, the S-faulty configuration is largely unaffected by the noise.

D Bit-flip Model vs. Kraus Operator Model

To evaluate how the noise models affect the protocol, we compared the behavior of a classical bit-flip noise model and a Kraus operator-based noise model. Although both approaches involve using the same error probabilities p_0 and p_1 and although they yield similar probabilities for individual measurement outcomes when simulated, their influence on the protocol's overall failure probability significantly differs.

For both of the noise models the probabilities can be shown as

$$p_0 = f_0|\alpha|^2 + (1 - f_1)|\beta|^2$$
$$p_1 = (1 - f_0)|\alpha|^2 + f_1|\beta|^2$$

.

The derivation from the Kraus operator based model can be found in the Appendix C.1. Even though, the bit-flip model applies the noise after the measurement, the resulting probabilities, since α and β are the probabilities of measuring state $|0\rangle$ and $|1\rangle$ respectively, turn out to be the same at the end of the process.

To investigate this further, we ran an experiment in which the four-qubit entangled state used in the protocol was generated and measured under either bit-flip noise or Kraus operator noise over 10,000 runs. The resulting post-measurement state distributions were consistent for both of the models as shown in the Table 1.

In order to dig deeper into where the noise models differ to produce different trends for the failure probability of the protocol, we designed another experiment where we applied noise either to only the sender or only the receivers. There was a key difference that in the bit-flip model, the effect of noise on either party resulted in similar failure probability trends. The results are shown in the Figure 10. In contrast, the Kraus operator model showed a significant failure probability increase only when the noise was applied to the receivers, not the sender.

Table 1: Measurement outcome probabilities and standard errors for Kraus Operator and Bit-flip Noise Models over 10,000 trials.

State	Kraus Operator	Bit-flip	SE Kraus Operator	SE Bit-flip
0000	0.0000	0.0000	0.000000	0.000000
0001	0.0022	0.0024	0.000469	0.000489
0010	0.0024	0.0024	0.000489	0.000489
0011	0.2973	0.2993	0.004571	0.004580
0100	0.0022	0.0022	0.000469	0.000469
0101	0.0751	0.0749	0.002636	0.002632
0110	0.0746	0.0738	0.002627	0.002614
0111	0.0238	0.0234	0.001524	0.001512
1000	0.0023	0.0022	0.000479	0.000469
1001	0.0743	0.0751	0.002623	0.002636
1010	0.0767	0.0746	0.002661	0.002627
1011	0.0234	0.0235	0.001512	0.001515
1100	0.2960	0.2960	0.004565	0.004565
1101	0.0236	0.0238	0.001518	0.001524
1110	0.0236	0.0237	0.001518	0.001521
1111	0.0026	0.0025	0.000509	0.000499

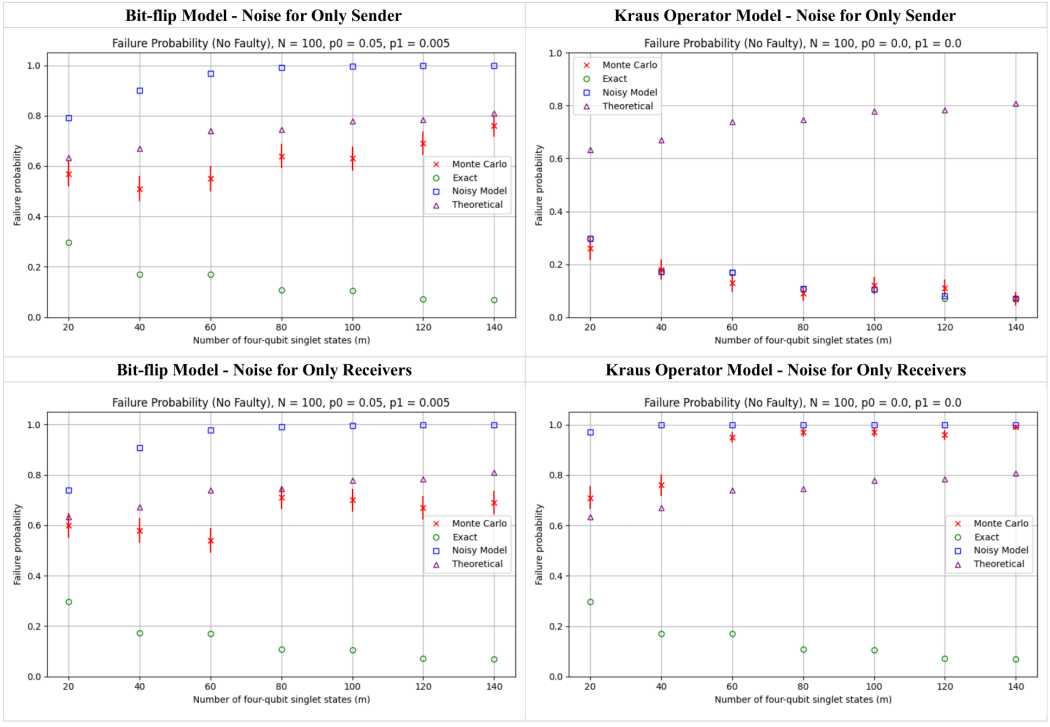

Figure 10: Failure probability applied to Sender and Receivers Separately.