CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Consensus via Anonymous Committees

Investigating Robustness Against Imprecise Security Settings and Quantum Adversaries

Master's thesis in Computer science and engineering

Olof Forsberg
Aron Arasan

Department of Computer Science and Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2026

Master's thesis 2026

Consensus via Anonymous Committees

Investigating Robustness Against Imprecise Security Settings and Quantum Adversaries

Olof Forsberg, Aron Arasan

UNIVERSITY OF
GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2026

Consensus via Anonymous Committees
Investigating Robustness Against Imprecise Security Settings and Quantum Adversaries

Supervisor: Elena Pagnin, Computer Science and Engineering
Examiner: Romaric Duvignau, Computer Science and Engineering

Master's Thesis 2026
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2026

Consensus via Anonymous Committees
Investigating Robustness Against Imprecise Security Settings and Quantum Adversaries
Olof Forsberg
Aron Arasan
Department of Computer Science and Engineering
Chalmers University of Technology and University of Gothenburg

Abstract

Current solutions for achieving agreement in distributed systems often scale poorly or are resource intensive. Achieving agreement is necessary to distribute correct public keys using key transparency logs. Consensus mechanisms that rely on anonymous committees are a novel approach with the explicit goal of improved scalability in this task. Anonymous committees are chosen using a cryptographic function, which allows them to prove membership. Nodes reveal committee membership only upon casting votes, and are therefore anonymous, making them difficult to target with attacks before voting. To improve scalability, it is possible to derive optimal settings from the system's assumed fractions of inactive and adversarial nodes.

This report quantifies how security is affected if those assumptions are incorrect. Using plots of bit security dependent on changes in inactive or malicious node shares it shows that the system is susceptible to such variations. Even using high security margins the system might be vulnerable when assumptions are only a few percentage points wrong. As the security is probabilistic, it does not break at any certain point. Instead security gradually weakens as the share of malicious nodes increases. Additionally, as future viability is necessary for adoption of new systems, suitable post-quantum secure cryptographic selection functions are presented and compared. As scalability is a major selling point of anonymous committees, it is relevant to find a measurement for how the size of the subset necessary to approve proposals changes as a function of node count. Testing data shows the subset size is only weakly dependent on the node count, and approaches an upper limit as node count increases.

Acknowledgements

We would like to thank our supervisor Elena Pagnin, for all her feedback and guidance throughout this thesis.

We would also like to thank Lucia Lavagnino, for her help and suggestions, and Paul Stankovski Wagner, for helping us get started with the Tail-Hammer library.

I, Aron, wish to thank my Lord and Saviour Jesus Christ, for His great love and care for me, and the abundant graces that He has poured over His unworthy servant.

Olof Forsberg and Aron Arasan, Gothenburg, 2026-07-04

Contents

List of Figures

List of Tables

1

Introduction

Modern society relies on digital communication and would come to a stop without it. Everything from bank transactions and messaging apps to most electronic devices in your home utilize it. A fundamental challenge in any digital communication is ensuring authentication, confidentiality and integrity. Authentication ensures that the sender is who they claim to be and that the receiver is the intended target. Confidentiality is information secrecy, that no one other than the sender and receiver read the contents. Integrity means that the information is unaltered after being sent. Without these safeguards, your bank transactions and other sensitive information might end up altered or fall into the wrong hands.

This poses the question, how can identity and trust be established in a digital setting? Most commonly this is established using signatures and certificates. These often rely on use of public key cryptography, where each party has a key pair. A public key which is widely distributed, and a private key which is never distributed. Cryptographic signatures verify the identity of the sender and that the message contents are unaltered. The private key is used to sign a message, and the public key is capable of verifying that the signature is correct. Certificates are cryptographic objects which bind identity to a public key. They generally contain an identity, a public key and a signature proving knowledge of the private key corresponding to the public key.

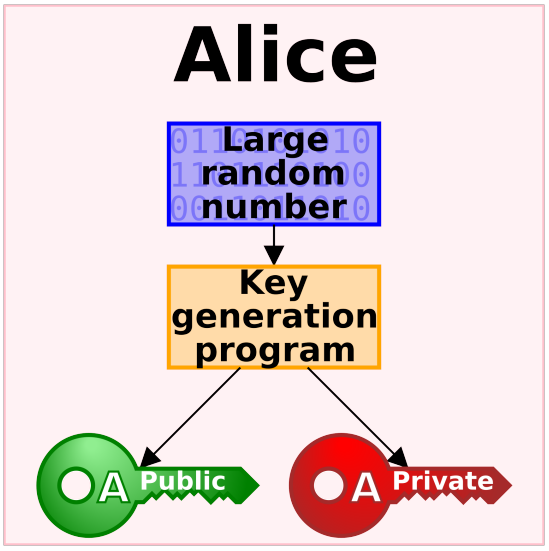


Figure 1.1: KohanX. Public-key generation. Wikimedia Commons (Public Domain).

For public-key systems to work, the public keys need to be readily available. A system for distributing public keys in an end-to-end setting is Key Transparency (KT). KT consists of a publicly verifiable append-only log of public keys and the parties who issued them, and a protocol which ensures consistent views across the network [1]. Append-only means no changes can be made to previously accepted identities and key combinations, only new additions are allowed. Updates to the log requires a common decision mechanism to ensure consistency. The protocol must detect illegal changes and ensure that all honest nodes, meaning all parties acting according to protocol, accept the same changes. For this there are multiple solutions.

The first of these are Certificate authorities (CAs) [2] which are real world entities that store and provide the certificates necessary to establish secure communication lines between parties. The trust derives from the existence of the real world entity and a history of providing correct certificates. However, reliance on these centralized entities is problematic, as it requires high trust in the CAs and also makes them obvious targets for attackers. The 2011 DigiNotar attacks are the most famous where domains such as Google, Microsoft and Mozilla were issued fraudulent certificates. This resulted in the bankruptcy of the company with the same name and the leakage of large amounts of emails.

Gossip style protocols [3] instead rely on nodes communicating peer-to-peer to disseminate information. Nodes randomly choose one or more neighbouring nodes to update regarding its current state. Upon receiving an update, a node applies a set of acceptance rules and may merge the new information into its own state. The system is fully decentralized with any node being able to disseminate updates and the implementation is relatively simple. Weaknesses are slow dissemination and weak guarantees for information consistency across the network. This is due to the fact that consistency checking is only done at random, not deterministically. The

system is also vulnerable to malicious nodes which can attempt to spread incorrect or inconsistent information.

A third alternative is blockchain systems [4] [5]. In a blockchain system all changes are made in a single, shared object, making differing views impossible after consensus is reached. Changes are organized into blocks which are linked back to their predecessors. Each block is selected through consensus, through a system called Proof of Work (PoW) or Proof of Stake (PoS). PoW means users have to demonstrate computational effort in producing the new block to get their share of votes while PoS means that a user has voting power proportional to their stake in the network currency. To establish a malicious KT log a majority of voting rights would need to be malicious, which is generally considered infeasible. For PoW, the weaknesses are the high computational requirements to create new blocks and high latency in block creation. As KT logs do not inherently define a notion of stake, a PoS-based approach would require introducing an external mechanism for assigning voting weight.

To address the limitations of poor scalability, excessive centralization and the limitations of blockchains, an additional approach to consensus has been proposed. Anonymous Committees (AC) are based on the idea of cryptographically choosing a subset of nodes to represent the entire system in a consensus protocol. A committee of nodes is chosen using a Verifiable Random Function (VRF), which also allows users to prove membership in the committee when voting. The nodes which are part of the committee are anonymous, meaning they do not reveal their committee membership until votes are sent. Therefore an adversary cannot target them specifically after being chosen. Using a subset of nodes drastically reduces the number of messages sent compared to the alternatives. Parameters for optimizing the scalability of ACs, while still achieving secure consensus, can be derived from the total node count and the fractions malicious and inactive nodes [6]. The quorum, the number of committee members that need to agree to approve a proposal, can be determined optimally under assumed fractions of malicious and inactive nodes.

ACs have been used in applications to achieve consensus. For instance, the Algorand blockchain uses PoS together with ACs in its consensus protocol [7], although the ACs here do not use optimal quorums based on malicious and inactive fractions. Another example is the proposed Consistency-or-Die (CoD) protocol [8]. CoD is a novel consistency protocol for KT, which uses an AC to reach agreement on updates to a KT log maintained by a central node.

As ACs are still quite new, there remains a list of research gaps. For example, as mentioned, the fractions of malicious and inactive nodes in the system can be used to optimize ACs. These two values are used to determine the optimal share of committee members necessary for a proposal to pass. However, in real-world scenarios, knowing the actual fractions is unreasonable. The implications of the actual fractions differing from the input fractions is yet to be investigated.

Another open research area is the post-quantum (PQ) security of AC selection processes based on VRFs. Quantum computers are expected to reach a level of computational power where many cryptographic assumptions become feasible to break within polynomial time [9]. Any developments within the area of cryptography

must take into account the possibility of quantum computers being able to break encryptions efficiently. Public-key systems are made especially vulnerable to quantum computing due to Shor's [10] algorithm, a quantum algorithm designed to solve both the prime factorization problem and the discrete logarithm problem. The difficulty of solving these problems have been leveraged in many cryptographic schemes since they are infeasible to solve for traditional computers.

In the CoD paper, it is stipulated that the VRF used must be unpredictable under malicious key generation [8]. In the two existing efficiency estimates, a VRF based on elliptic-curve cryptography is used [11][8]. These are based on assumptions that are not post-quantum secure and may become vulnerable in a future with large-scale quantum computers [9]. For selecting nodes using a VRF to be possible in the future, a suitable post quantum VRF needs to be found. It is also relevant to discern whether changing to a post-quantum secure VRF would have any adverse effect on system function. For example, if epoch duration increases greatly for current systems such as CoD.

As scalability is the main advantage of ACs, understanding how the required number of votes scales with network size is pertinent. This is relevant to how the system scales when the amount of nodes can increase into billions and even further. Having a defined measurement would be a useful tool when comparing AC-based solutions to other alternatives.

1.1 Purpose

The intention of this project is to further the understanding of ACs by answering three questions.

- Question 1: How is the system affected when the assumed fractions of malicious and inactive nodes are incorrect?
- Question 2: Is there a suitable post-quantum VRF for use in AC selection? Would there be any adverse effect on the function of ACs from using a post-quantum secure VRF?
- Question 3: What is the quorum size order of growth as a function of the node count?

1.2 Scope and Limitations

An early design choice in relation to Question 1 was to use an existing simulation framework for estimating quorum sizes under varying system parameters. The framework takes as input parameters such as the fractions of malicious and inactive nodes and was modified to serve the purposes of this study.

For Question 2, constructing a new Verifiable Random Function (VRF) was outside the scope of this thesis due to the complexity of cryptographic design and implementation. Instead, existing standardized or proposed VRF constructions were eval-

uated to determine their suitability under post-quantum constraints. If no suitable construction had been identified, this would have been reported as a limitation of the study.

For Question 3, deriving quorum growth purely analytically was considered infeasible within the scope of this thesis due to the complexity of the underlying probabilistic model. Instead, a simulation-based approach was used to generate empirical data from which conclusions were drawn. This approach introduces a dependency on the correctness of the implementation.

1.3 Outline

- Chapter 2: Technical background and details. Necessary for someone who is not familiar with cryptography and distributed systems.
- Chapter 3: Related work. Detailing other similar systems and principles. Not directly necessary to understand our work, but more the competing alternatives or inspirations.
- Chapter 4 Assumed fractions: Detailing the process, results and relevant discussion for question 1, Assumed fractions
- Chapter 5: Post Quantum VRF: Detailing the process, results and relevant discussion for question 2, Post Quantum VRF
- Chapter 6: Detailing the process, results and relevant discussion for question 3, quorum order of growth dependent on network size.
- Chapter 7: Conclusion. Summary of findings. Discussion involving findings from multiple questions.

2

Background

This section explains relevant technical details regarding the problem presented earlier.

2.1 Distributed Systems

A distributed system [12] consists of multiple computing devices, called nodes, that cooperate to solve a common task. This allows the system to utilize the combined computational resources of all participating nodes. However, distributing tasks introduces several challenges in the form of communication overhead, implementation complexity, and the need to ensure consistency between all nodes.

For example, a network of nodes is tasked with running a large computation that can be split into multiple smaller problems. If the problem is sufficiently large, then splitting it across multiple nodes will speed up execution by a factor approximately equal to the number of nodes, ignoring communication overhead. While distribution is a powerful tool, it also introduces challenges. Examples include nodes disconnecting, messages being lost in transit, and nodes behaving as adversarial agents.

Another common use is for servers. If a website or service is to be available worldwide, then multiple servers are generally needed, both to share the load of incoming traffic and to reduce latency. At the same time, an update received by one server must eventually be propagated to all others. For a banking system, if two servers simultaneously receive a payment request and approve it, then more money than is available in the account can be withdrawn. This is called double-spending and is a common problem that distributed systems managing transactions need to solve.

2.1.1 Consistency

Consistency [13] describes the guarantees a distributed system provides regarding the visibility and ordering of updates across nodes. Intuitively, it determines whether different nodes observe the same state of the system. In the banking example, consistency means that all nodes agree on the balance of each account.

One common definition is strong consistency, where every read operation returns the result of the most recent completed write. Consequently, concurrent reads from

different nodes will return identical results. Strong consistency provides the abstraction of a single up-to-date system image, even though the data is distributed across multiple nodes.

Weaker consistency models also exist. For example, eventual consistency guarantees only that updates accepted by one correct node will eventually be propagated to all other correct nodes. During this propagation period, different nodes may return different results for the same query. Such guarantees may be insufficient for systems that require a globally consistent view of data, such as a KT-log system.

An inconsistency can occur if updates are lost, discarded, or applied differently across nodes. Some distributed systems address such situations by temporarily employing stronger coordination or consensus mechanisms to re-establish a common system state before resuming normal operation.

2.1.2 Safety and security

Safety [12] in distributed systems refers to the property that nothing bad ever happens during execution. More formally, a safety property guarantees that a system never reaches an incorrect or undesirable state at any finite point in time. Examples of safety violations include:

- Revealing secret information
- Violating data consistency guarantees
- Producing incorrect or inconsistent results

Safety properties are closely related to guarantees such as consistency, since violations of consistency often stem from violations of safety. In practice, many attacks on distributed systems first aim to violate consistency, which may then enable further exploitation such as unauthorized data access or control over system behavior.

Within cryptography, similar ideas are captured by the notion of security. A common framework is information security, which considers three main properties: confidentiality, integrity, and authentication. Confidentiality ensures that an adversary cannot learn secret information, such as bank account details. Integrity ensures that data cannot be modified without authorization. Authentication ensures that entities are correctly identified and cannot impersonate one another.

Cryptographic security is typically quantified using computational assumptions rather than absolute guarantees. One common notion is bit security, where a system is said to have b bits of security if the best known attack requires approximately 2^b computational effort. Larger values of b therefore correspond to exponentially stronger security guarantees, making successful attacks computationally infeasible in practice.

2.1.3 Liveness

While safety means protection against bad events, such as attacks influencing or gaining information from the system, liveness [12] means that something good eventually happens. Formally, good means that the system eventually makes progress with its task. For example, a new view of system state is accepted and distributed. A system can be considered safe, while making no progress, if it does not leak information or produce faulty results.

Violations of liveness would be that the system enters an infinite loop, inability to reach consensus, deadlock or starvation. Unless the system has a self-stabilizing mechanism these problems mean that progress will stall. For the system this means that its main task is left unattended.

Bit security can also be used to measure the likelihood of liveness violations. Generally bit security in relation to liveness has significantly lower demands as a break of the property does not give away any secret information. As such very low bit securities is viable as it only needs to be unlikely, not cryptographically secure under brute-force attacks.

2.1.4 Participating nodes

Relevant to the question of assumed fractions is the concept of malicious nodes, also known as byzantine nodes [14]. They act arbitrarily, in violation of the protocol or actively attempt to bring down the system. Nodes can act either independently or as a cohesive unit, with the latter case being more harmful. Using their power as nodes they try to gain influence and reveal secret information.

In contrast there are honest nodes. They have no hidden agenda and are acting as expected, fulfilling their role in the protocol. The third category of nodes are inactive nodes. Nodes which have been part of the network but are not any longer or fail to respond to contact. Inactive nodes have no direct effect on the system but might influence quorum sizes and skew the share of malicious nodes among active nodes.

2.1.5 Consensus

In situations where a system of nodes must agree on updates to a shared or distributed resource, a consensus protocol is required. Consensus [12] is the property that enables a distributed system to reach agreement on a common state or decision despite the absence of a central authority. By ensuring that all honest nodes eventually agree on the same sequence of updates, consensus allows distributed systems to maintain a consistent view of shared data.

Achieving consensus in a distributed environment is challenging due to factors such as network delays, message loss, node failures, and malicious behavior. Nodes may have only partial knowledge of the system state and must communicate over unreliable networks, making it difficult to determine whether another node has failed or

is merely experiencing delays. As a result, consensus protocols must be designed to tolerate faults while still allowing the system to make progress.

Consensus must remain achievable even in the presence of malicious nodes that actively attempt to prevent or manipulate agreement. Such nodes may send conflicting information to different participants, refuse to follow the protocol, or attempt to approve invalid state transitions. These actions can threaten the two primary goals of a consensus protocol: safety and liveness. Safety ensures that honest nodes do not reach conflicting decisions and that invalid updates cannot be accepted by the system. Liveness ensures that the protocol continues to make progress and that valid updates are eventually processed and agreed upon.

2.2 Anonymous Committees

Anonymous Committees (ACs) can be used to improve the scalability of consensus protocols, while also providing protection from adversaries wishing to corrupt important nodes in the protocol. An AC is selected by randomly choosing a small subset of all nodes to be members of the committee. The members of the committee are then tasked with executing some part of the protocol, without disclosing their identities before they have finished their task. The selection of a small committee provides scalability, while the undisclosed membership of the committee means that an adversary does not know who to target to influence the protocol. The number of selected members for the committee, or the committee size, can be fixed or probabilistic. [6].

2.2.1 AC Selection for ACs with Probabilistic Sizes

This project considers ACs in which each node has an equal probability of being selected into the committee. This means that every node has the same probability p of being selected into the committee. If there are a total of N nodes, then the committee has an expected size of pN.

Protocols utilizing ACs with this kind of selection usually do so with the help of a Verifiable Random Function (VRF) [6]. A VRF [15] is a function that produces a random-looking output, along with a proof which verifies that the output was generated by that VRF and by a specific entity. In systems using VRFs for AC selection, each node locally runs their VRF and retrieves an output along with the aforementioned proof. The output then determines whether that node is part of the committee, and the output together with proof can be used to convince other nodes of one's committee membership.

The way this works is that each node runs the VRF with their own private key and a common input shared by all nodes. This will output a value r and a proof π_r. This value is then compared with a system wide threshold value T, e.g checking that $r < T$. If $r < T$, then the node is part of the committee. Later on, the node can reveal that it is on the committee by publishing (r, π_r). With the proof π_r and the public key of node claiming to be on the committee, other nodes can verify that

r is truly the result of the VRF with the common input. The probability of being selected into the committee can be defined as $p = T/2^{\ell}$ where ℓ is the range of the VRF's outputs [6].

2.2.2 Quorums

A problem in ACs is that of identifying the minimum number of participants needed to execute the protocol to guarantee that enough honest members have participated. This is particularly challenging in ACs with probabilistic sizes, since then it is not possible to know how many members are in the committee. For scalability, it is desirable that this number be as small as possible. However, it also needs to be sufficiently large to ensure that malicious nodes can not determine the outcome. We call this value the *quorum*. When receiving messages from Q committee members, a node in the system can be certain that enough honest participants have executed the protocol, except with negligible probability [6].

2.3 Verifiable Random Functions

While VRFs have been described in simple terms in Chapter 1 and Section 2.2.1, this section provides a formal and more extensive definition of VRFs. A VRF [15] is a function that produces a pseudorandom output along with a proof which verifies that the output was generated by that VRF. A pseudorandom output is deterministically generated from a secret key and input, but is computationally indistinguishable from a truly random value to anyone without knowledge of the secret key.

VRFs consists of a few different procedures [8]:

- $KeyGen(\lambda) \rightarrow (pk, sk)$: Generate a secret key (sk) and a public key (pk) based on security parameter λ
- $Prove(sk, s) \rightarrow (r, \pi_r)$: Generate a value r and a proof π_r based on sk and a seed s
- $Verify(pk, s, r, \pi_r) \rightarrow 0$ or 1: Given pk, s, r and π_r, check that π_r is a proof that r was generated by s and the secret key paired with pk

A VRF must fulfill these properties:[16]:

- Uniqueness: no values $(pk, s, r_1, r_2, \pi_1, \pi_2)$ can satisfy $Verify(pk, s, r_1, \pi_{r_1}) = Verify(pk, s, r_2, \pi_{r_2}) = 1$ when $r_1 \neq r_2$
- Provability: if $Prove(sk, s) \rightarrow (r, \pi_r)$, then $Verify(pk, s, r, \pi_r) = 1$
- Pseudorandomness[15][16]: for any probabilistic-polynomial-time (PPT) adversary $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ playing the following experiment:
  1. $(pk, sk) \leftarrow KeyGen(\lambda)$
  2. $(x, st) \leftarrow \mathcal{A}_1^{Prove(\cdot)}(pk)$
  3. $(v_0, \pi_0) \leftarrow Prove(sk, x)$

4. $v_1 \xleftarrow{\$} \{0,1\}^{|v|}$
5. $b \xleftarrow{\$} \{0,1\}$
6. $b' \leftarrow \mathcal{A}_2^{Verify(\cdot)}(v_b, st)$

a VRF is pseudorandom if $\Pr[b = b' | \mathcal{A}$ runs the experiment above$] \leq \frac{1}{2} + negl(\lambda)$

A VRF has a lifetime before it needs to be reset. This is the number of evaluations, the number of times it can perform the prove operation before security guarantees regarding uniqueness and unpredictability break down.

2.3.1 Malicious Key Generation and Unbiasability

A requirement for any VRF used in AC selection is unpredictability under malicious key generation [8] [17]. This property ensures that the output of the VRF remains unpredictable, regardless of how a malicious user chooses the keys [18]. If a VRF without this property is used, then it is possible that a malicious node can choose input keys such that the VRF output is more likely to admit them into the committee. This property is also captured by what is called unbiasability in [15].

2.3.2 Elliptic Curve VRF

The current implementations of CoD use an Elliptic Curve-based VRF (ECVRF) [8]. Elliptic Curve Cryptography is used to generate key pairs using the equation $y^2 = x^3 + ax + b \mod p$ [19] where p is a large prime.

The system relies on the hardness of the Elliptic Curve Discrete Logarithm Problem (ECDLP). Given a generator point g on the curve, a secret scalar x and public key $A = x * g$, it is computationally infeasible to determine x from knowing A, g.

This assumption has been deemed reliable for a long time but is now threatened by advances within quantum computing. Shor's Algorithm [10] for quantum computers can break the ECDLP in polynomial time. At some point when quantum computers become sufficiently effective this will make the problem feasible to solve, something which motivates the need for new, quantum secure alternatives.

2.3.3 Post-Quantum Secure Assumptions

Here we will introduce the different assumptions and overarching systems which are deemed post-quantum secure and can be used for VRFs.

2.3.3.1 Lattice

Lattice-based cryptography is based upon the hardness of problems defined over mathematical lattices and is the most researched post quantum approach [20]. A lattice is a discrete set of points within an n-dimensional coordinate system. Given

a basis vector for each dimension each point in the lattice is some combination of these.

$v = a_1 * b_1 + a_2 * b_2 + a_3 * b_3...$

The hardness lies in problems such as the shortest vector problem (SVP) and closest vector problem (CVP). A key concept is that lattices can be represented by different bases. The original lattice has basis vectors which are nearly orthogonal, which simplifies finding the combination of basis vectors to reach a certain point. A "bad" set of basis vectors are nearly parallel, which make them essentially interchangeable, forcing an exhaustive search of alternatives.

The task for the receiver is to find point in the new lattice closest to that point using the nearly parallel basis vectors while the sender has the "good" basis vectors. While cases up to three dimensions can be visualized, the problem often uses hundreds of dimensions to increase the number of possible combinations. The exhaustive search is deemed to be computationally infeasible, even for quantum computers.

2.3.3.2 Hash

Hash-based cryptography relies on the properties of cryptographic hash functions, assumptions such as preimage resistance, second preimage resistance and collision resistance. A collision means that two inputs x_1 and x_2 produce the same output y. Preimage resistance means that calculating using x to find y is quick, finding x using y should be computationally infeasible. Second-preimage resistance means given an input x_1, finding a different input x_2 such that the hashes of x_1 and x_2 are equal, is computationally infeasible. These assumptions are not based upon a single mathematical problem, but on the empirical difficulty of breaking well-designed hash functions. Given a sufficiently large function output, a brute-force attack should require an infeasible amount of time, even for quantum computers.

2.3.3.3 Isogenies

Isogeny-based cryptography is a relatively recent development and therefore a less proven extension of the assumptions on elliptic curves [21]. Given multiple elliptic curves, it is assumed to be computationally infeasible to find an isogeny between any two. This results in relatively small key sizes compared to alternatives. An isogeny is a structure-preserving map between elliptic curves $\phi : E \to E'$.

In 2022 [22] a paper proved the most famous isogeny-based system to be insecure. It detailed an attack which broke the Supersingular Isogeny Diffie-Hellman (SIDH) assumption in polynomial time. The attack exploited previously unaccounted structural properties, showing that the underlying hardness assumption was weaker than expected. This made SIDH unusable and forced redesigns for most isogeny-based systems. Since then most new isogeny-based systems are further developments which solves the specific problem in some way.

2.4 Optimal Quorums

A novel security analysis of ACs with probabilistic selection was presented in [6], which can be used to determine the optimal quorum, given the total population size and the fractions of malicious and inactive nodes. The general idea is to link the tail areas of binomial distributions to the probability of detrimental events. These probabilities can in turn be calculated with arbitrary precision using the Python library provided by the same paper. The paper focuses on random committees with probabilistic size, for which each user has the same probability of being selected. The analysis is also applicable to non-anonymous committees, but the context of the results is that of ACs [6], which is also the focus of this project.

In more detail, let N be the total number of nodes, and p the probability of being selected as members of the AC. The total number of nodes selected for the AC can then be modeled as a random variable that follows the binomial distribution $B(N, p)$. If $N = H + M + I$, with H, M and I being the number of honest, malicious and inactive nodes respectively, then $B(N, p) = B(H + M + I, p) \overset{d}{=} B(H, p) + B(M, p) + B(I, p)$, with $\overset{d}{=}$ signifying that the two sides have identical distribution functions [6]. This implies that the selection of honest and malicious users into the committee can be analyzed independently, with $B(H, p)$ and $B(M, p)$ signifying distinct distributions if $H \neq M$.

More broadly, let $S, L \in \mathbb{N}$ such that $S < L$, let p be the committee selection probability, and let $\mathcal{B}_X \sim B(X, p)$, for $X \in \mathbb{N}$. The *split-bell approach* considers the probabilities:

$$\begin{cases} \Pr[\mathcal{B}_L < f_L(Q)] \\ \Pr[\mathcal{B}_S \geq f_S(Q)] \end{cases} \tag{2.1}$$

Where $Q \in \mathbb{N}$, and $f_L, f_S : \mathbb{N} \setminus \{0\} \to \mathbb{N}$. The first probability is called the right bell, and the second the left bell. The right bell can generally be interpreted as the probability of liveness breaking, while the left bell the probability of safety breaking. The general idea is that the quorum must be determined such that these two probabilities are small enough.

As a concrete example, let the probabilities be instantiated by $L = N$, $S = M$ and $f_L(x) = f_S(x) = x$. This yields the system.

$$\begin{cases} \Pr[\mathcal{B}_N < Q] \\ \Pr[\mathcal{B}_M \geq Q] \end{cases} \tag{2.2}$$

If the first probability is small, then the probability of there not being enough committee members to reach a quorum is small. If the second probability is small, then the probability of the malicious committee members being able to reach a quorum on their own is small. For the right bell, total population N can be replaced with the active population $A = N(1 - f_I)$, if the system has inactive nodes ($f_I > 0$).

In this way, through different instantiations of L, S, f_L and f_S, quorums ensuring that the probability of different bad events are small enough can be found.

2.4.1 Quorums and Shadow Committees in Asynchronous Networks

Depending on the network conditions, a quorum must satisfy different conditions. In particular, in an asynchronous network an adversary may arbitrarily suppress or delay messages. An adversary could then theoretically choose to let some subset of the committee votes propagate to some part of the system, while letting some other subset of the committee votes propagate to another part of the system. If such an adversary collaborates with the malicious nodes selected into the committee, then having a quorum which makes the left bell of 2.2 small is not enough to ensure safety. An adversary could split the honest nodes in the committee in half and influence them so that they vote for two conflicting states. Then, the malicious users on the committee could vote for both of these states. If this results in both states receiving a number of votes that is larger than or equal to the quorum value, then some honest users in the system may believe that one consensus has been reached, while others may believe another consensus has been reached [6].

To prevent this, the expressions above can be instantiated with $L = N, S = M + \frac{H}{2}$, and $f_L(x) = f_S(x) = x$, which yields the following bell system:

$$\begin{cases} \Pr[\mathcal{B}_N < Q] \\ \Pr[\mathcal{B}_{M+\frac{H}{2}} \geq Q] \end{cases} \tag{2.3}$$

Finding a quorum that satisfies these conditions implies that the probability of there being too few committee members, and the probability of the aforementioned attack, are both small. A quorum Q such that the two bells of 2.3 are small is called an 'at least one honest party" quorum, since together they guarantee that any subset of Q committee members will have at least one honest committee member, and that two Q-sized subsets of the committee can not vote for different views [6].

2.4.2 Tail Measures

Tail measures takes the probabilities of a system, and a quorum, and returns a unified measurement of the security of the system. More formally, a tail measure is a function $\tau : \mathbb{N} \setminus \{0\} \to [0, 1]$, which takes a quorum and outputs a probability. For two-bell system according to System 2.1, a tail measure is parametrized by the selection probability p, the populations L and S of the binomial distributions of the two bells, and the two quorum-scaling functions f_L and f_S.

There are two tail measures presented in [6] that are relevant in this project. The first is the max tail measure:

$$\tau_p^{(S,L,f_S,f_L)}(q) = \max\left\{\Pr\Big[\mathcal{B}_S \geq f_S(q)\Big],\ \Pr\Big[\mathcal{B}_L < f_L(q)\Big]\right\} \tag{2.4}$$

The second tail measure is the safety-over-liveness tail measure:

$$\tau_p^{(S,L,f_S,f_L)}(q) = \begin{cases} \Pr[\mathcal{B}_L < f_L(q)] & \text{if } \Pr[\mathcal{B}_S \geq f_S(q)] \leq \Pr[\mathcal{B}_L < f_L(q)] \\ 1 & \text{otherwise} \end{cases} \tag{2.5}$$

With a tail measure $\tau_p^{(S,L,f_S,f_L)}$, the quorum optimal Q can be rigorously defined as:

$$Q = \min\{q \in [0, L] \mid \tau_p^{(S,L,f_S,f_L)}(q) \leq 2^{-b}\} \tag{2.6}$$

in which b is the bit security of the system.

2.4.3 Approximation Formulas for Optimal Quorums

Approximation formulas for the optimal quorum Q and the associated selection probability p can be derived from the normal approximation of the binomial distribution [6].

In the 'at least one honest party" case with the max tail measure, the following formulas can be used to calculate approximate values for Q and p in an asynchronous network:

$$p = \frac{\phi_b^2}{\left(\sqrt{N} - \sqrt{(1 + f_M)/2}\right)^2 + \phi_b^2}$$

$$Q = pN\sqrt{(1 + f_M)/2}$$

Here, ϕ_b is a quantile of the normal distribution that satisfies $\Pr[\mathcal{N} \leq \phi_b] = 2^{-b}$, for the desired bit secutity b. The details and derivation of these equations can be found in [6].

2.5 Application of ACs: Consistency Protocol for Key Transparency

Key Transparency (KT) is a distributed system which can be used to create a link between public keys and their issuer [8]. KT consists of a data structure containing the public keys, and a consistency protocol ensuring that all nodes have access to the same data structure, and that they all agree on the history of changes of the structure [1]. The data structure can be called a KT log [8]. The purpose of the

consistency protocol in KT is that any pair of honest users will always accept the same KT log.

Consistency-or-Die (CoD) [8] is a novel protocol that was proposed due to the available solutions to the KT consistency problem being unsatisfactory in large-scale systems [1]. It is meant to protect key transparency logs against split-view attacks by not relying on a vulnerable or inefficient consensus. Upon detecting any inconsistency in the key logs all honest nodes shut down, i.e. the nodes are either consistent or dead, thereby the name. The system chooses to sacrifice liveness for security.

2.5.1 Split-View Attacks

Split-view attacks are based on desynchronizing the system which is done by making at least two nodes view system state differently [8]. For example, making two nodes accept different views of the system state. This breaks the assumption of a single, consistent view of the system. If this has been achieved a multitude of other attacks become possible as security and integrity guarantees break down. Most security guarantees rely on that all honest nodes view the system identically.

A split-view attack against a KT-log works by compromising the KT-server, from which users then receive different KT-logs. The users will then view the system differently, achieving a split view. To protect KT-logs against split-view attacks there needs to be a system for verifying updates to KT-logs. For this task consensus algorithms are used as the system needs to agree on updates.

2.5.2 The Phases of the CoD Protocol

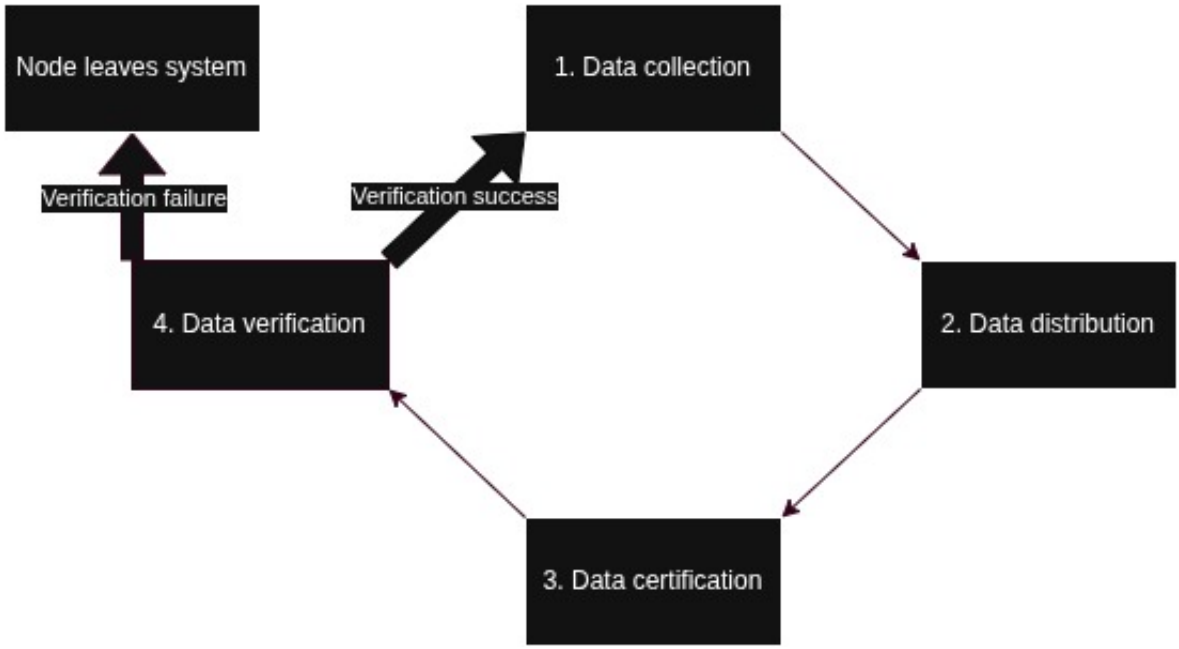
The nodes in CoD can have a few different roles:

1. Channel maintainer ($\mathcal{CM}$): untrusted central node
2. Users: Nodes that download data from the $\mathcal{CM}$. The list of users may change over time.
3. Publisher: can send data to the $\mathcal{CM}$. Both users and external parties may have this role.
4. Endorsers: a subset of Users who audit the information received from the $\mathcal{CM}$. Elected via a VRF and updated for each epoch.

By design CoD is asynchronous, meaning there is no set clock or timeout for certain actions. The time unit is instead epochs, which is the basic loop of the system. Each epoch is split into four phases:

- Collection: Users send new keys to the $\mathcal{CM}$ which collects them to create an updated log $View_e$. It also contains a proof for that $View_e$ contains all the keys from the last log, as well as any new proposals.
- Distribution: The $View_e$ is distributed by the $\mathcal{CM}$ to the users

- Certification: A subset of users, endorsers, verify the key log and send their signed endorsements to the $\mathcal{CM}$ which collects the responses
- Verification: $\mathcal{CM}$ sends the final certificate containing endorsements to all users. If (i) a quorum of endorsers supports it and (ii) it aligns with the previously accepted KT-log, the updated KT-log is accepted. Any nodes which deem it not align with the previous KT-log become inactive.



2.5.3 Cryptographic Sortition in CoD

During the certification phase, a set of endorsers are cryptographically chosen to form an AC, which is then tasked with verifying the KT log created by the $\mathcal{CM}$ in the collection phase. This is where the protocol's VRF is used. Each node has a keypair $(sk_{\text{VRF}}, pk_{\text{VRF}})$ generated by *KeyGen*. The $Prove(sk_{\text{VRF}}, s_e)$ procedure is run by each node with their own private key and with a shared input s_e. Each node then compares the output y with a common threshold T. If $y < T$, then the node is chosen as a member of the committee. Each chosen committee member is tasked with verifying that the KT log has not been maliciously handled by the $\mathcal{CM}$. If verification succeeds, the committee member endorses the received view by signing it. Together with the signed endorsement, the node also sends the output y of the VRF, together with the proof that y was generated by the VRF [8]. By doing this, the endorser can prove that it was in fact randomly chosen to be part of the committee.

2.5.4 Malicious nodes in CoD

In CoD, malicious behavior can manifest as attempting to manipulate votes on new KT logs. For instance, a malicious publisher may propose replacing the public key of another node with another belonging to a malicious node. If sufficiently many nodes are malicious, the update is passed. Nodes will then believe they are

communicating with the original node, but messages will actually go to the impostor. This constitutes an impersonation or key substitution attack.

When an anonymous committee is selected, it may contain a subset of malicious nodes. The system is compromised if the number of malicious nodes in the committee reaches or exceeds the quorum threshold. Therefore, the system sets its quorum size based upon expected fraction of malicious nodes. The goal is that the bit security is at least 128, i.e. that the probability of such quorums is 2^{-128}.

2.5.5 Quorums in CoD

There are a few bad events that may occur that would lead to breaking the consistency of CoD. One of these is the event that the committee does not have enough honest nodes to reach a quorum. For the probability of breaking the consistency of CoD to be negligible, the probability of this bad event must also be negligible. The paper calls this probability $negl_{\text{Qrm}}(N, f_M, f_I)$ [8].

The negligibility of $negl_{\text{Qrm}}(N, f_M, f_I)$, and thus the security of CoD, hinges on the quorum [8]. N is the total number of nodes, f_M the fraction of malicious nodes and f_I the fraction of inactive nodes. If H and M are the number of honest and malicious nodes selected for the committee for some epoch, then the quorum Q is chosen such that

$$\Pr[\mathcal{B}_{M+\frac{H}{2}} \geq Q] \leq \Pr[\mathcal{B}_H < Q] = negl_{\text{Qrm}}(N, f_M, f_I) \tag{2.7}$$

The bit security can then be interpreted as $b = -\log_2(\Pr[\mathcal{B}_H < Q])$ [8]. The negligibility of the term $negl_{\text{Qrm}}(N, f_M, f_I)$ at any point of time is only guaranteed if the actual population size, fraction of malicious nodes and fraction of inactive nodes are in fact equal to N, f_M and f_I.

As a final remark, CoD can not handle a malicious fraction $f_M \geq 1/3$, as then it becomes impossible to find a suitable quorum. If $f_I = 0$ and $f_M = 1/3$, then $M = \frac{1}{3}N$ and $H = \frac{2}{3}N$, which implies that $M = \frac{1}{2}H$. This then means that $M + H/2 = H/2 + H/2 = H$. This means that the two random variables $\mathcal{B}_H$ and $\mathcal{B}_{M+\frac{H}{2}}$ are identical, and CoD is then trying to set a quorum Q such that $\Pr[\mathcal{B}_H < Q]$ and $\Pr[\mathcal{B}_H \geq Q]$ both become negligible, which is contradictory. Thus the maximum malicious ratio of CoD is $f_M^{max} = 1/3$ [11].

3

Related work

The problem of ensuring consistency and disseminating updated information in distributed systems has been studied extensively since the inception of distributed computing. Numerous constructions have therefore been proposed, targeting different trust models, scalability requirements, and application domains. Here multiple other solutions, both for KT and general consensus, are presented. Understanding other solutions is context for why this work is relevant, but not necessary to understand the process or the results.

System	Consistency mechanism	Scalability (qualitative)	Intended task
Certificate transparency	Central log	Medium	Key transparency
CONIKS	Gossip-based propagation	High	Key transparency
OPTIKS	Aggregator verification	High	Key transparency
ETHIKS	Blockchain consensus	Medium	Key transparency
Algorand	VRF-based anonymous committee	High	Consensus
PBFT	Byzantine agreement	Low	Consensus

Table 3.1: Comparison of representative consistency mechanisms in distributed systems. Scalability is given as a qualitative assessment based on bottlenecks such as centralization, computational loads and communication costs. If no such limitations were found scalability is listed as high. In cases where only minor limitations exist it is listed as medium. If a strong limitation, or multiple minor exists, scalability is listed as low.

3.1 Certificate Transparency

Certificate Transparency [23] is a public log system for TLS certificates designed to detect incorrect or malicious certificates. It was introduced as a response to incidents such as the 2011 DigiNotar attack, where a trusted certificate authority issued fraudulent certificates enabling man-in-the-middle attacks.

Certificate Transparency is based on publicly auditable, append-only logs of certificates. Certificate authorities submit issued certificates to these logs, which are implemented as Merkle tree-based data structures. This enables efficient verification of inclusion and consistency proofs.

Clients receiving a certificate can verify that it has been logged correctly via signed certificate timestamps (SCTs), while external monitors continuously audit logs to

detect inconsistencies or misbehavior. The system therefore relies on transparency and external verification rather than a single trusted authority.

Due to its reliance on centralized logging infrastructure and auditing services, Certificate Transparency scales less effectively to highly decentralized or extremely large-scale systems, but provides strong consistency guarantees under its trust assumptions.

3.2 YOSO

YOSO (You Only Speak Once) [24] is a system model for distributed computation designed to reduce communication complexity and improve privacy properties in multi-party protocols.

Instead of iterative interaction between participants, YOSO restricts each participant to a single communication round. Nodes perform local computation and send a single message to an aggregator, which combines inputs and produces a final output.

This reduction in interaction complexity improves efficiency and reduces exposure of intermediate state information. However, the model introduces reliance on an aggregation phase, which may become a bottleneck or trust assumption depending on the implementation.

To mitigate this, many YOSO-based constructions incorporate additional verification mechanisms or distributed aggregation strategies.

3.3 CONIKS

CONIKS [25] is a key transparency system designed to reduces reliance on trusted third parties for public key distribution.

CONIKS uses a Merkle tree-based directory structure to maintain a publicly verifiable mapping between user identities and public keys. Each user can independently verify inclusion of their key in the directory through cryptographic proofs.

The system supports privacy-preserving verification by allowing users to monitor their own entries without revealing unnecessary information to external observers. Updates to the directory are appended in a verifiable manner, enabling auditability and detection of inconsistencies. Inconsistencies are generally found gossip-style communication where nodes compare their views of the system state, which is a weaker guarantee than consensus for accepting updates

CONIKS reduces reliance on certificate authorities but still depends on directory maintainers for update propagation and consistency management.

3.4 OPTIKS

OPTIKS [26] is a later extension of CONIKS designed to improve scalability and reduce communication overhead in key transparency systems.

OPTIKS introduces a computation model inspired by YOSO, where much of the computational burden is shifted toward precomputation and aggregation phases. Instead of requiring all nodes to participate in every verification step, OPTIKS uses intermediate aggregators that combine and validate partial results.

The protocol can be summarized in the following phases for participating nodes:

1. Local computation of private input
2. Precomputation of verification data
3. Submission of computed values
4. Verification of aggregated output
5. Reception of final result

Aggregators perform message collection, combination, and verification of correctness before distributing results.

While OPTIKS improves scalability compared to CONIKS, it introduces reliance on aggregation infrastructure, which must be designed carefully to avoid centralization bottlenecks.

3.5 ETHIKS

ETHIKS [5] combines the CONIKS key transparency design with blockchain-based consensus mechanisms, specifically those used in Ethereum [4].

The system stores updates to a Merkle prefix tree on a blockchain, ensuring that all modifications are globally ordered and publicly verifiable. This guarantees strong consistency properties, since all participants observe the same history of updates once consensus is reached.

A key advantage of this design is the ability to trace all historical updates and detect forks or inconsistencies in state evolution. However, this comes at the cost of increased computational and communication overhead associated with blockchain consensus.

As a result, ETHIKS is generally considered more suitable for smaller-scale deployments where strong auditability is required but scalability constraints are less severe.

3.6 Algorand

Algorand [17] is a distributed consensus protocol used in the cryptocurrency of the same name. It achieves consensus through the use of VRF-based sortition to select

anonymous committees that propose and vote on blocks.

Each participants probability of being selected is weighted according to their stake in the system, corresponding to an Honest Majority of Money (HMM) assumption. This implies that security depends on the assumption that a majority of economic stake remains honest.

The protocol can also be instantiated under an Honest Majority of Users (HMU) assumption, although this typically results in reduced resistance to Sybil-style attacks.

Algorand serves as a major inspiration for committee-based consensus systems such as CoD, particularly in its use of cryptographically selected anonymous committees.

3.7 PBFT

Practical Byzantine Fault Tolerance (PBFT) [27] is a classical Byzantine fault-tolerant consensus protocol designed to tolerate malicious or faulty replicas in an asynchronous network setting.

The protocol operates under a replicated state machine model, where a primary node proposes an ordering of requests and backup replicas validate and agree on this ordering through a three-phase process: pre-prepare, prepare, and commit.

Safety is achieved under the assumption that at most f out of $3f + 1$ replicas are faulty, ensuring that any two quorums of size $2f + 1$ intersect in at least one honest node. This guarantees consistency across correct replicas.

While PBFT provides strong consistency guarantees, its communication complexity scales quadratically with the number of nodes, limiting its applicability in large-scale decentralized systems.

4

Assumed Fractions

In this chapter, we present the results of our investigation of the robustness of AC systems with quorums calculated with Tail-Hammer. That is, we endeavor to understand how the actual security of these systems are affected by incorrect assumptions in fractions of malicious and inactive nodes. This is done both for ACs in general, and for CoD, a practical application of such a system.

The question is split into two cases:

1. Measuring security when $f'_M > f_M$
2. Measuring security when $f'_I \neq f_I$

We did not measure the security in the case when $f'_M < f_M$. The security can only increase in this case, and since the quorum is already optimized for the desired bit-security level, this additional security is superfluous. It would instead be more interesting to measure efficiency. Since the actual fraction f'_M is smaller than the expected fraction f_M, there should exist a quorum $Q' < Q$ with which the same bit security can be achieved. However, in the interest of time, this is not done here.

The chapter is outlined as follows: first, a description of the general methodology, and then the results for ACs in general, and the results for the AC used by CoD. The first part explains, in general terms, the manner in which the bit security is measured as a function of changes in the malicious or inactive fraction is presented. This general methodology encompasses the methodology behind the results for ACs in general (Section 4.2), and the results for the AC used by CoD (Section 4.3).

4.1 General Methodology

The high-level idea is to compute the precise optimal quorum Q for some population, bell system and tail measure, and then to consider and measure how the security is affected when the population is altered.

In more detail, we first choose a starting population, a bell system, a tail measure and the target bit security. A population is defined by (N, f_M, f_I), with N being the population size, and f_M, f_I being the fraction of malicious and inactive nodes respectively. A bell system, defined according to 4.1, and a tail measure $\tau_p^{(S,L,f_S,f_L)}$, are also chosen. These are instantiated with some values for L and S, depending on the population.

$$\begin{cases} \Pr[\mathcal{B}_L < f_L(Q)] \\ \Pr[\mathcal{B}_S \geq f_S(Q)] \end{cases} \tag{4.1}$$

After setting these, the optimal quorum Q, with respect to these settings and achieving the target bit security, is computed with code based on the Tail-Hammer library [28]. After this, the original population is changed to (N, f'_M, f'_I). This altered population yields new values for L and S, let them be called L' and S'. This in turn yields a new bell system:

$$\begin{cases} \Pr[\mathcal{B}_{L'} < f_L(Q)] \\ \Pr[\mathcal{B}_{S'} \geq f_S(Q)] \end{cases} \tag{4.2}$$

However, instead of calculating the optimal Q' for this new system, the old Q is inserted into 4.2. This yields the probabilities of whatever bad events are associated with the two bells for Q on the population (N, f'_M, f'_I). These are then converted into the bit security of the system with respect to the measure $\tau_p^{(S',L',f_S,f_L)}$. This is then the bit security of the system when the quorum was optimized for the fractions f_M, f_I, but the actual fractions happen to be f'_M, f'_I. This is repeated for several populations, and the resulting bit securities are used to plot the bit security of the original setup as a function of f'_M or f'_I.

The code used to calculate and plot the bit securities can be found here [29]. The quorums are calculated using 1000 bits of precision. For all the plots in this chapter, the bit security at 210 different values for Δf_M or Δf_I are calculated and plotted. For the Δf_M case, these 210 values are evenly distributed in the interval $\Delta f_M \in [0, 0.15]$, and in the interval $\Delta f_I \in [-0.15, 0.15]$ for the inactive case. Along with this, in some cases cutoff points are calculated. For example, in the $f'_M > f_M$ case in Section 4.2.1, the largest value for Δf_M such that the new bit security $b' \geq 80$ is calculated. These cutoff values are calculated exactly. In the example, the exact malicious population H' is found such that the bit security with H' is ≥ 80, but with $H' + 1$ it becomes < 80.

4.1.1 Considerations on Modeling the Changes of Fractions

One additional point that must be considered is how a population (N, f_M, f_I) is altered when a fraction f_M or f_I is changed. For example, consider the case when one part of the population is altered such that it is larger than it was originally. How should the other parts of the population be made smaller, while keeping the same N as the total population size? This requires decisions to be made on how to model these cases.

Given a population (N, f_M, f_I), the number of inactive, malicious and honest nodes in the population can be calculated as: $I = f_I N$, $M = f_M N$ and $H = N - I - M$. We only perform alterations that preserve the total population, i.e $H' + M' + I' = N$ is always true. Thus, when the original population is altered to (N, f'_M, f'_I), the new

number of inactive, malicious and honest nodes become: $I' = f'_I N$, $M' = f'_M N$ and $H' = N - I' - M'$.

In the case that $f'_M > f_M$, the most natural option may be to simply reduce H in equal proportion to how much larger M becomes. Intuitively, one might think that increasing M and decreasing H is always the worst-case scenario. For the asynchronous networks presented in Section 2.4 with $I > 0$, this is actually not the worst case. The worst case here is actually increasing M, and reducing I instead of H. This is because the left bell $\Pr[\mathcal{B}_{M+\frac{H}{2}} \geq Q]$ has a slower rate of growth if M is increased and H is decreased in equal proportion, compared with the case of only increasing M and keeping H constant. The right bell $\Pr[\mathcal{B}_A < Q]$, however, will actually decrease since the total active population is larger. However, when using either the max of tails or the safety-over-liveness tail measures, this latter fact has no impact on overall security. In CoD, the situation is slightly different, with reducing H and reducing I both being equally bad. This is due to the fact that the right bell is instead $\Pr[\mathcal{B}_H < Q]$, which will decrease if H decreases.

In this project, we chose the natural modeling option for $f'_M > f_M$: only M and H are changed, while I is unchanged. This provides a clean separation between the cases of $f'_M > f_M$ and $f'_I \neq f_I$, while also modeling the worst-case scenario for CoD.

As for modeling the $f'_I \neq f_I$ case, again it is not straightforward. What does it mean for the actual active population to be larger or smaller? Does the honest or malicious part of the actual population have more or fewer active nodes?

The decision made in this project was to keep the actual malicious population constant, while letting the actual honest population grow or decrease as a function of f'_I. That is, $f'_H = f_H - (f'_I - f_I)$. This means that when there are more inactive nodes, there are fewer honest nodes, and when there are fewer inactive nodes, there are more honest nodes. This allows for a clean separation between the cases of $f'_M > f_M$ and $f'_I \neq f_I$. There are two additional reasons for modeling this case in this way. The first is simple: the worst-case scenario when $f'_I > f_I$ is obviously keeping $f'_M = f_M$ while decreasing f'_H. The second reason is that it is not intuitively clear what happens when $f'_H > f_H$. One might instinctively conclude that a larger honest population should exclusively lead to a higher level of security. However, recall that the probability of a split-view occurring in an asynchronous network is $\Pr[\mathcal{B}_{M+H/2} \geq Q]$. This means that a larger H and a constant M would lead to an increase of this probability. This is an interesting quirk of ACs in asynchronous networks, and is thus worthwhile to model.

4.2 Optimal Quorums for Anonymous Committees

In this section, we consider the case when subcommittees are required to have at least one honest party in asynchronous networks. The target bit security is 128, and the quorum Q is optimized with respect to the max of tails measure. This means that the bell system of 4.1 is instantiated as:

$$\begin{cases} \Pr[\mathcal{B}_A < Q] \\ \Pr[\mathcal{B}_{M+\frac{H}{2}} \geq Q] \end{cases} \tag{4.3}$$

and the tail measure $\tau_p^{(S,L,\mathbf{id},\mathbf{id})}$, with **id** being the identity function, is instantiated as:

$$\tau_p^{(M+\frac{H}{2},A,\mathbf{id},\mathbf{id})}(q) = \max\left\{ \Pr[\mathcal{B}_A < q],\ \Pr[\mathcal{B}_{M+\frac{H}{2}} \geq q] \right\}$$

At one point, the safety-over-liveness measure was chosen to match the measure of Table 4 in the Tail-Hammer paper [6]. However, this is not good fit for our purposes, especially in the $f'_M > f_M$ case. This is because $\Pr[\mathcal{B}_A < q]$ is unaffected by any changes in the fraction of malicious nodes, while $\Pr[\mathcal{B}_{M'+\frac{H'}{2}} \geq q]$ increases with increasing f'_M. Since the optimal quorum Q is chosen such that $\Pr[\mathcal{B}_{M+\frac{H}{2}} \geq q] \approx \Pr[\mathcal{B}_A < q] < 2^{-b}$, this means that increasing the left-side probability will almost immediately lead to $\Pr[\mathcal{B}_{M'+\frac{H'}{2}} \geq q] > \Pr[\mathcal{B}_A < q]$. From equation 2.5, one can see that this in turn implies that the safety-over-liveness measure will return 1. This means that the bit security becomes 0 nearly instantaneously when increasing f'_M above f_M, which obviously means that any more detailed measurements or plots are superfluous. The max of tails measure does not have this property, and is thus the tail measure used for both $f'_M > f_M$ and $f'_I \neq f_I$ for ACs with 'at least one honest party" quorums.

4.2.1 $f'_M > f_M$

This is a theoretically straightforward case. We have the following equalities: $M' = f'_M N$, $\Delta M = M' - M$ and $H' = H - \Delta M$. The left bell of 4.3 becomes $\Pr[\mathcal{B}_{M'+\frac{H'}{2}} \geq Q]$, with $M' + \frac{H'}{2} = \Delta M + M + \frac{H-\Delta M}{2} = M + \frac{H}{2} + \frac{\Delta M}{2}$, which is larger than $M + \frac{H}{2}$ for $f'_M > f_M$. Thus the left-bell probability will strictly increase as f'_M increases.

However, it is not straightforward to theoretically argue that the bit security goes below some particular value at some exact Δf_M. For this, computing the bit security as a function of Δf_M, as explained in Section 4.1, and computing the exact Δf_M cutoff, provides some insight. This can be seen in Figure 4.1.

Figure 4.1 illustrates this case for the initial population ($N = 10^9, f_M = 0.3, f_I = 0$). The strict decrease of the bit security as Δf_M increases is in line with the theoretical considerations. Beyond this, one can also observe that the bit security becomes 80 just after Δf_M is ≈ 0.0666.

Repeating this for populations with $f_M = 0.15$ and $f_M = 0.01$ and adding the results to figure 4.1, provides a way to compare how the same absolute change in the malicious fraction affect security among the different populations. This is done in figure 4.2. The cutoff Δf_M for bit security $b = 80$ does grow with decreasing initial f_M, but not by much. Going from $f_M = 0.3$ to $f_M = 0.01$ leads to the cutoff

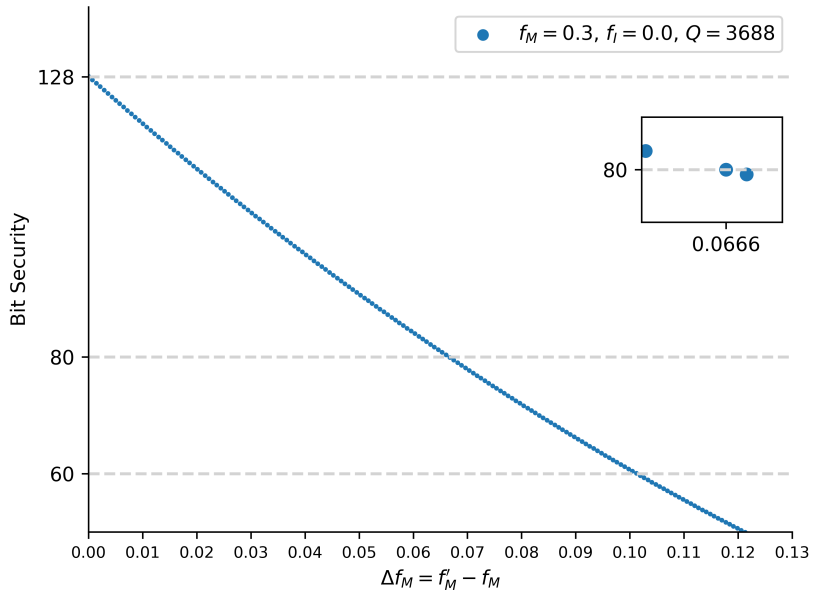


Figure 4.1: Bit security as a function of $\Delta f_M = f'_M - f_M$, for increasing $f'_M > f_M$. The 'at least one honest party" quorum Q is optimized for 128 bits of security on the population $(N = 10^9, f_M = 0.3, f_I = 0.0)$, with respect to the max tail measure. The box to the right shows the largest Δf_M such that the bit security is ≥ 80.

going from $\Delta f_M \approx 0.06660$ to only $\Delta f_M \approx 0.08679$, despite the initial malicious fraction being 30 times smaller.

4.2.2 $f'_I \neq f_I$

Recall from Section 4.1 that $\Delta f_I > 0$ means that $H' < H$, while $\Delta f_I < 0$ means that $H' > H$, while the malicious population is always M. When $\Delta f_I > 0$, the active population becomes smaller, which means that the probability $\Pr[\mathcal{B}_{A'} < Q]$ grows. However, since $H' < H$, the opposite of the previous section occurs: $\Pr[\mathcal{B}_{M'+\frac{H'}{2}} \geq Q]$ decreases. Thus, $\Delta f_I > 0$ implies that the liveness of the system decreases, while the safety increases.

On the other hand $\Delta f_I < 0$ implies that the liveness of the system increases, while safety decreases. $\Delta f_I < 0$ obviously implies $A' > A$, which increases liveness. However, $H' > H$ means that $\Pr[\mathcal{B}_{M'+\frac{H'}{2}} \geq Q] > \Pr[\mathcal{B}_{M+\frac{H}{2}} \geq Q]$, and thus safety decreases.

Figure 4.3 shows the results for $(N = 10^9, f_M = 0.1, f_I = 0.2)$, with respect to the max of tails measure. The plot confirms the counter-intuitive result, in line with our theoretical considerations: $\Delta f_I < 0$ leads to increased liveness but decreased safety, and when $\Delta f_I > 0$ the opposite occurs. Figure 4.4 plots the two bells separately, which illustrates this even more clearly.

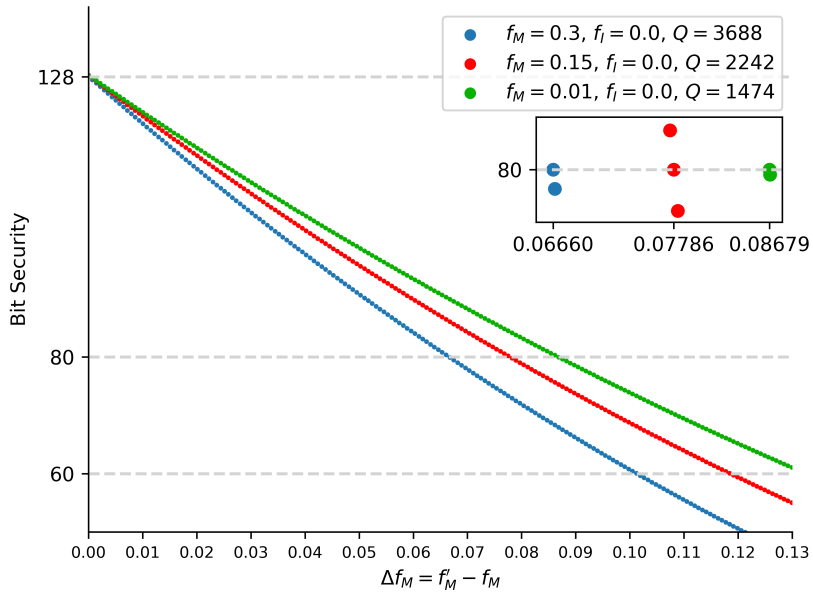


Figure 4.2: Bit security of three systems as a function of $\Delta f_M = f'_M - f_M$, for increasing $f'_M > f_M$. The 'at least one honest party" quorums are optimized for 128 bits of security on each population with respect to the max of tails measure. The zoomed in image adds the cutoff values.

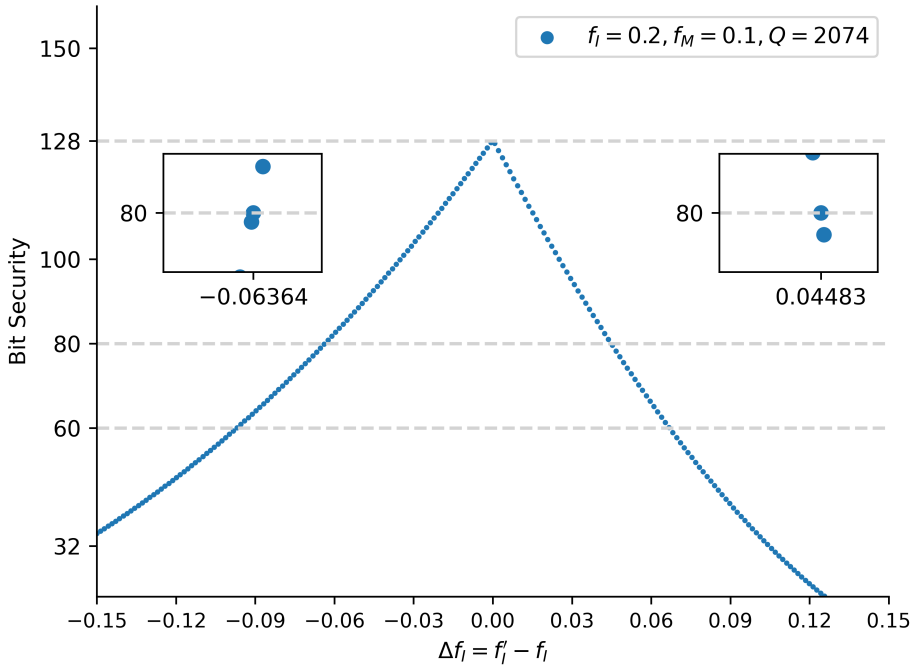


Figure 4.3: Bit security as a function of $\Delta f_I = f'_I - f_I$, for $f'_I \neq f_I$. The 'at least one honest party" quorum Q is optimized for 128 bits of security on the population $(N = 10^9, f_M = 0.1, f_I = 0.2)$, with respect to the max tail measure. The zoomed in boxes show the two cutoff values for Δf_I

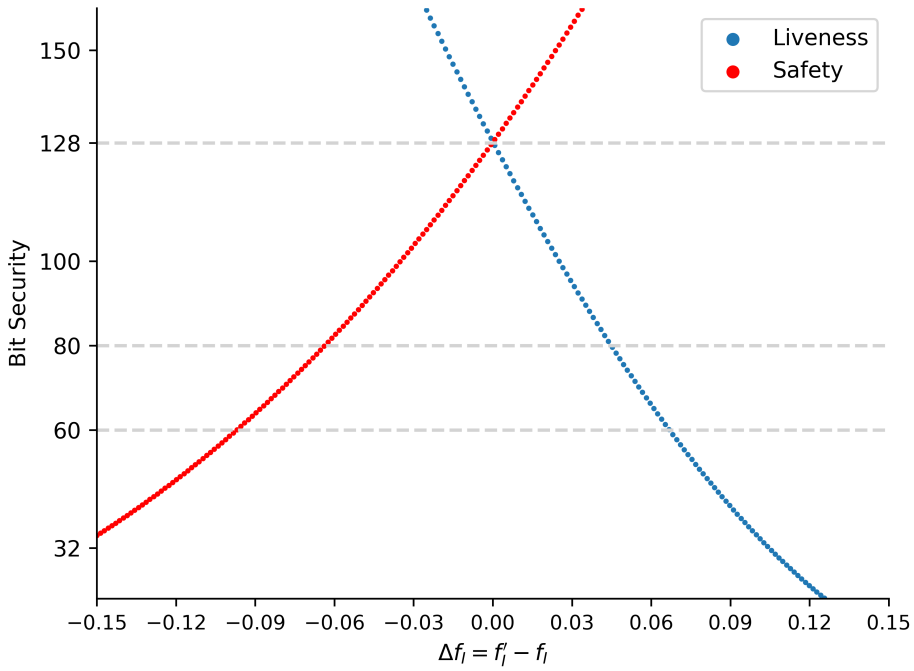


Figure 4.4: Bit security of liveness and safety as a function of $\Delta f_I = f'_I - f_I$, for $f'_I \neq f_I$. The 'at least one honest party" quorum Q is optimized for the same population as in Figure 4.3, for 128 bits of security on the population ($N = 10^9$, $f_M = 0.1$, $f_I = 0.2$).

4.3 Quorum for the Committee in Consistency-or-Die

This section investigates the AC used in CoD. Recall from Section 2.5.5 that an optimal quorum Q in CoD is the smallest $Q \in \mathbb{N}$ that satisfies the following chain of inequalities:

$$\Pr[\mathcal{B}_{M+H/2} \geq Q] \leq \Pr[\mathcal{B}_H < Q] \leq 2^{-b}$$

It turns out that this is almost equivalent to finding a quorum for the 'at least one honest party" case with the safety-over-liveness tail measure in the asynchronous network setting. The only difference is that $L = H$, instead of $L = N$, resulting in a more stringent liveness constraint. We call this kind of quorum simply the CoD quorum.

$$\begin{cases} \Pr[\mathcal{B}_H < Q] \\ \Pr[\mathcal{B}_{M+\frac{H}{2}} \geq Q] \end{cases}$$

$$\tau_p^{(M+\frac{H}{2},H,\mathbf{id},\mathbf{id})}(q) = \begin{cases} \Pr[B_H < q] & \text{if } \Pr[\mathcal{B}_{M+\frac{H}{2}} \geq q] \leq \Pr[\mathcal{B}_H < q] \\ 1 & \text{otherwise} \end{cases}$$

The target bit security b was in this case chosen to be 256 instead of 128, due to CoD being susceptible to grinding. With these settings, the procedure is the same as that of Section 4.2.

4.3.1 $f'_M > f_M$

Due to the different liveness constraint, this case is actually a bit different compared with the $f'_M > f_M$ case of ACs in general. That is because in this case $f'_M > f_M$ affects liveness. Using the same notation as in Section 4.2.1, we have that $H' = H - \Delta M$ and $M' + \frac{H'}{2} = M + \frac{H}{2} + \frac{\Delta M}{2}$. The derivative of the term $M' + \frac{H'}{2}$ with respect to ΔM is $\frac{1}{2}$, while for the term H' it is -1. Thus, intuition would suggest that $\Pr[\mathcal{B}_H < Q]$ actually increases faster than $\Pr[\mathcal{B}_{M+\frac{H}{2}} \geq Q]$ when increasing f'_M.

For a quorum optimized for $(N = 10^9, f_M = 0.01, f_I = 0.2)$, Figure 4.5 shows that the bit security of the system goes below 128 at around $\Delta f'_M \approx 0.06878$. Plotting the two bells separately shows that it is in fact liveness that has a higher rate of decrease with respect to Δf_M, and that it hits the 128-bit threshold first. Safety on the other hand, reaches 128 bits later at $\Delta f_M \approx 0.09403$.

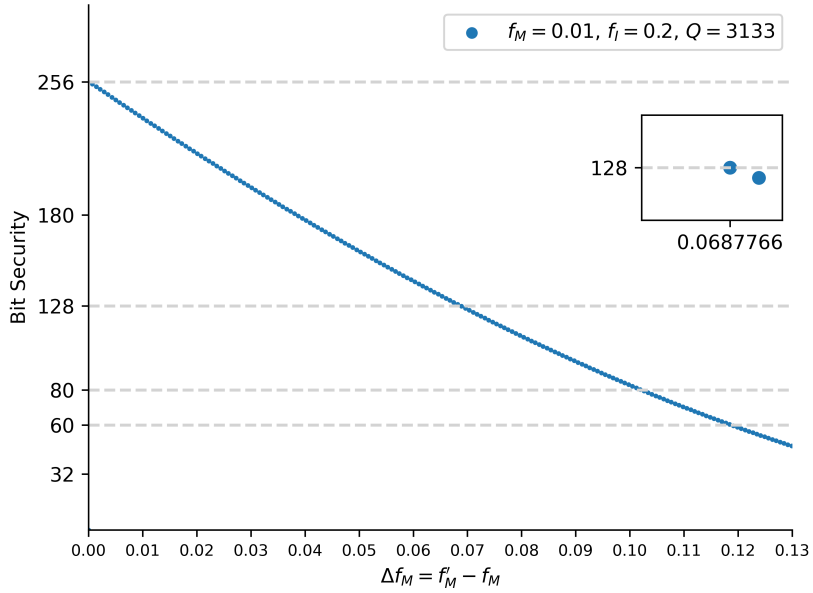


Figure 4.5: Bit security as a function of $\Delta f_M = f'_M - f_M$, for increasing $f'_M > f_M$. The CoD quorum Q is optimized for 256 bits of security on the population $(N = 10^9, f_M = 0.01, f_I = 0.2)$, with respect to the safety-over-liveness measure. The zoomed-in box shows the last Δf_M value where the bit security is still ≥ 128.

Adding curves for $f_M = 0.1$ and $f_M = 0.2$ shows $f_M = 0.2$ yields a much steeper curve, which can be seen in figure 4.7. Already at $\Delta f_M \approx 0.01575$, the bit security has already reached 128. At $\Delta f_M = 0.06$, the bit security becomes approximately 0.00107.

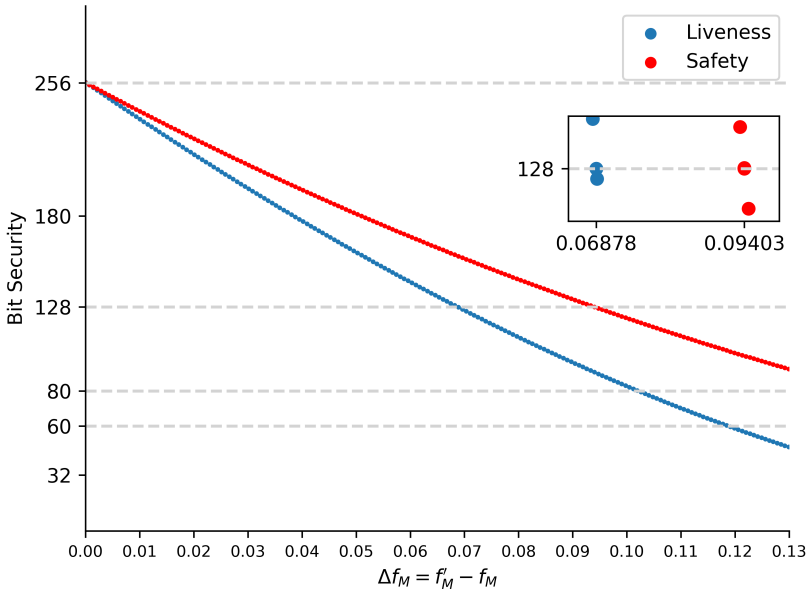


Figure 4.6: Bit securities of safety and liveness as a function of $\Delta f_M = f'_M - f_M$, for increasing $f'_M > f_M$. The CoD quorum Q is optimized for 256 bits of security on the population ($N = 10^9, f_M = 0.01, f_I = 0.2$), with respect to the safety-over-liveness measure. The zoomed-in box shows the last Δf_M value where the bit security is still ≥ 128.

Our interpretation is that this is due to the approaching of the maximum malicious ratio (see Section 2.5.5). When $f_I = 0$, this ratio is 1/3, or 1/3 of the active population. When $f_I > 0$, the maximum malicious ratio f_M^{max} is found by solving the equation $\frac{M}{A} = \frac{f_M^{max} N}{(1-f_I)N} = \frac{f_M^{max}}{1-f_I} = \frac{1}{3}$. With $f_I = 0.2$, this yields $f_M^{max} = \frac{4}{15} \approx 0.267$. This would then explain why the bit security is almost 0 at $\Delta f_M = 0.06$: here, $f'_M = 0.26$ is very close to f_M^{max}.

4.3.2 $f'_I \neq f_I$

While the details may be different, the general relationship between Δf_I and the bit security is the same in the case for CoD as it is for ACs with 'at least one honest party" quorums. The safety constraint is the same, while liveness depends on $\mathcal{B}_H$ instead of $\mathcal{B}_A$. However, due to our modeling decisions, when $\Delta f_I < 0$, both $H' > H$ and $A' > A$. When $\Delta f_I > 0$, we instead have $H' < H$ and $A' < A$. Thus, the liveness of both systems decreases at $\Delta f_I > 0$, and increases at $\Delta f_I < 0$.

However, the safety-over-liveness measure does not work well for our purposes in this case. Similar to Section 4.2, when $\Delta f_I < 0$ liveness increases while safety decreases.

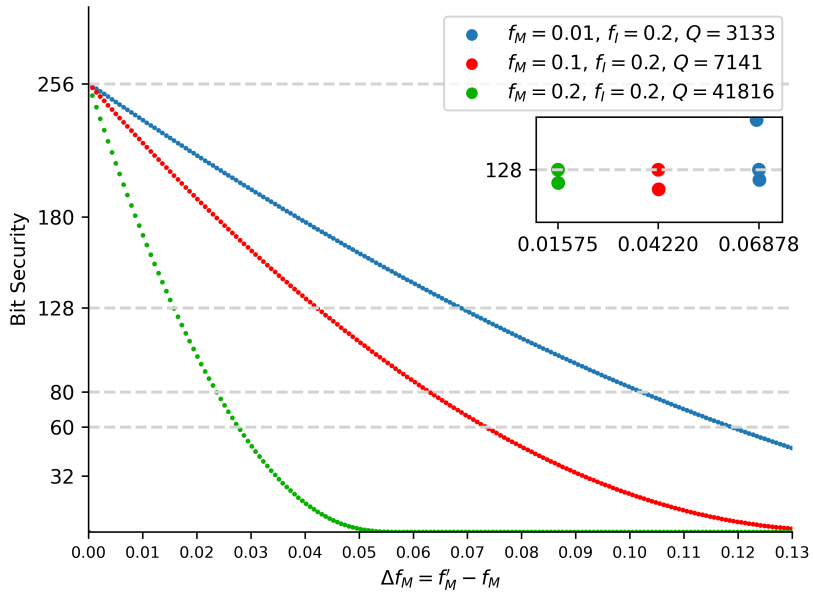


Figure 4.7: Bit security of three CoD systems as a function of $\Delta f_M = f'_M - f_M$, for increasing $f'_M > f_M$. The CoD quorums are optimized for 256 bits of security on each population with respect to the safety-over-liveness tail measure. The zoomed in image adds the cutoff values.

Thus, the bit security goes almost immediately to 0 under this tail measure. This can be seen in Figure 4.8. Plotting safety and liveness separately visualizes the case better, as seen in Figure 4.9

4.4 Discussion

The $f'_I \neq f_I$ case is in some ways the more troublesome one, since the system is affected by any change in the inactive population. This is different from the $f'_M \neq f_M$ case. In the latter case, with exact knowledge of the actual f_M, there is no need to worry about security decreasing if that fraction decreases over time. However, the same does not hold for f_I. If the inactive population ever decreases, then security will also decrease.

Instead of having perfect knowledge of the actual fraction of malicious users, a designer could set f_M to be the maximum permissible fraction. In this case, the system does not become less secure if $f'_M < f_M$. However, the same is not possible with f_I. When selecting the quorum for an AC, the designer can not, in the same way, set f_I to be the maximum permissible fraction of inactive users in the system, and expect that an inactive fraction lower than f_I poses no threat.

It is also clear that application-specific details matter a lot, even if they may be seemingly minor. The clearest example of this in our results can be found in the $\Delta f_M > 0$ case for CoD quorums. In this system, liveness is dependent on the honest

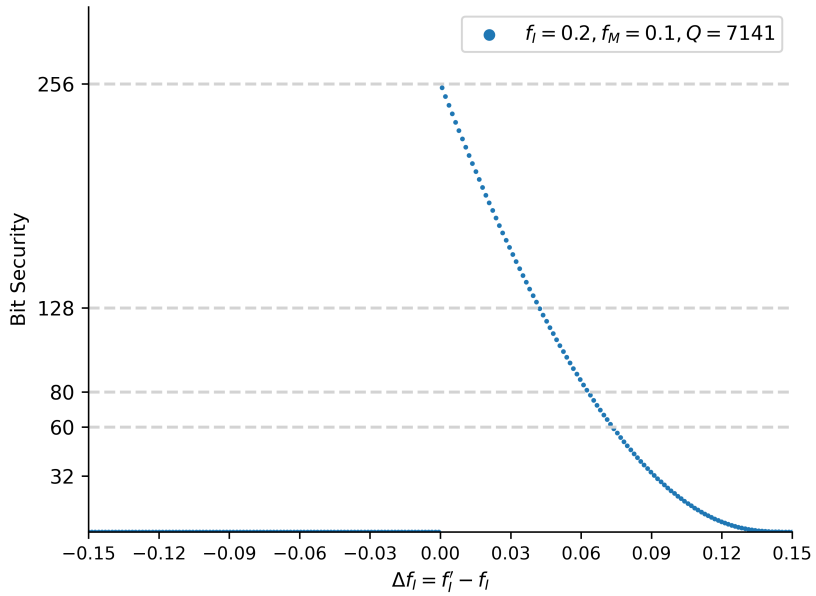


Figure 4.8: Bit security as a function of $\Delta f_I = f'_I - f_I$. The CoD quorum Q is optimized for 256 bits of security on the population ($N = 10^9, f_M = 0.1, f_I = 0.2$), with respect to the safety-over-liveness measure.

population rather than the larger active population. As explained in Section 4.3.1, this means that $\Delta f_M > 0$ reduces liveness more than it reduces safety. This is quite different from the case with 'at least one honest party" quorum, in which liveness is completely unaffected by Δf_M. Thus, an ostensibly small difference in the bell system leads to a pretty significant security implication. This could mean that the 'at least one honest party" quorum results in section 4.2 are not widely applicable, even though the intention was for this to be a more general case. Instead it may only be applicable to systems with the exact same quorums. However, similar analysis on other AC applications would be needed before making any such conclusion.

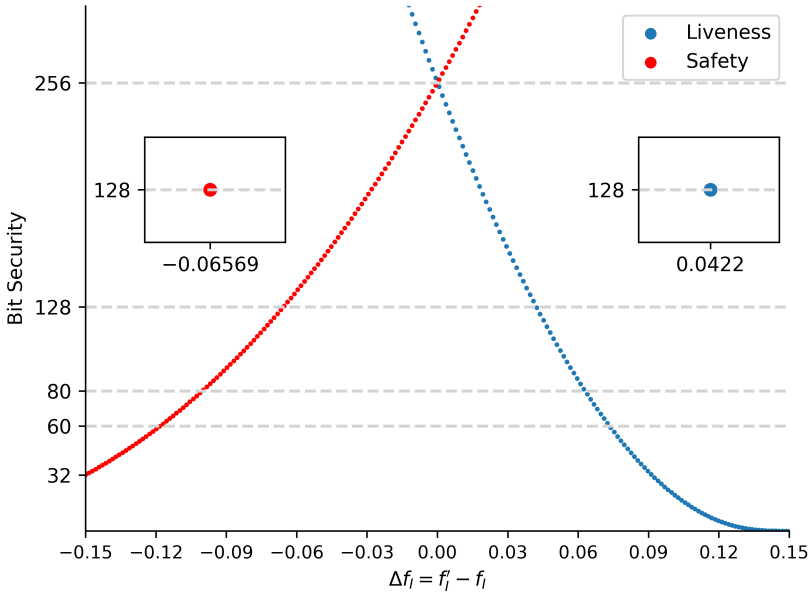


Figure 4.9: Bit security of liveness and safety as a function of $\Delta f_I = f'_I - f_I$, for $f'_I \neq f_I$. The CoD quorum Q is optimized for the same population as in figure 4.8, for 256 bits of security on the population ($N = 10^9, f_M = 0.1, f_I = 0.2$).

5

Identifying a Suitable VRF for Post-Quantum Secure AC Systems

Here, the findings regarding post-quantum secure VRFs suitable for ACs are presented. This means the criteria and merits which were determined, the alternatives found, comparisons with determined criteria and finally comparing overall suitability in relation to each other.

5.1 Post-Quantum VRF

To find suitable post-quantum secure VRFs for AC systems, an outline defining the necessary properties was created. The key requirements were found to be resistance to quantum algorithms as well as unbiasability. Unbiasability means that no adversary can increase the likelihood of a favorable outcome through key selection. This means all nodes are as likely to be part of any group chosen using VRF output. All security guarantees regarding choosing subsets of nodes will be severely weakened or broken if adversaries can make themselves more likely to be chosen.

Beyond fulfilling these properties, eligible constructions would have keys and proofs that are as small as possible, and rely on strong cryptographic assumptions. Smaller keys and proofs will reduce network load and transit times. If multiple alternatives are deemed to be sufficiently secure then epoch speed using them is a distinguishing factor.

VRF name	LB-VRF [15]	LaV [20]	X-VRF [30]	**iVRF** [31]	**DeuringVRF** [21]	ECVRF [32]
Security	Lattice	Lattice	Hash	Hash	Isogeny ($OMIP_{2dim}$)	Elliptic Curve
Post-Quantum	Yes	Yes	Yes	Yes	Yes	No
Unbiasable	Yes	Unsure	Unsure	Yes	Yes	Yes
# of Evals	1	2^{128}	2^{18}	2^{18}	Unlimited	Unlimited
PK	3.32 kB	$\approx$ 5.81 kB	64 B	32 B	192 B	32 B
SK	0.45 kB	Unsure	132 B	Unsure	Unsure	32 B
Value	84 B	124 B	Unsure	0 B	Unsure	32 B
Proof	4.94 kB	$\approx$ 10.27 kB	2.76 kB	608 B	256 B	80 B

Table 5.1: Merged comparison of VRF schemes. Some values or properties were not possible to find in their reports and have as such been left as "Unsure". Values from ECVRF are taken from [30][31]. Values marked as unlimited are not truly infinite, but large enough to not warrant a self-imposed upper limit. Other systems will automatically refresh keys after the limiting number of evaluations has been reached. The iVRF value size is marked as 0 B as it can be derived from the proof and does not need to be transmitted separately.

5.1.1 Post-Quantum VRF alternatives

In this part we will introduce the VRFs which have been surveyed from the literature and are assumed to be post-quantum secure.

5.1.2 LB-VRF

LB-VRF [15] is regarded as the first post-quantum VRF and is based on two standard lattice hardness assumptions, Module Short Integer Solution (MSIS) and Module Learning With Errors (MLWE).

MSIS [33] is a generalisation of the Short Integer Solution to module lattices. It is defined over modules of rank d over a polynomial ring.

Let R be a ring of integers of a number field and let $R_q = R/qR$ for a modulus $q \geq 2$. For integers $m, d \geq 1$ let

$$\mathbf{A} = (\mathbf{a_1} \dots \mathbf{a_m}) \in R_q^{d \times m}$$

The goal is to find a short, non-zero vector z such that:

$$\mathbf{Az} = \mathbf{0} \pmod{q}$$

The MSIS assumption states that finding such a solution is computationally hard for appropriately chosen parameters.

MLWE is also a generalisation, but of the LWE problem. It comes from an equation system

$$\mathbf{b} = \mathbf{A} * \mathbf{s} + e$$

where e is a small error, sampled from a noise distribution [34]. Given a list of such systems solving for the secret s is assumed to be computationally difficult. LB-VRF

[15] prioritizes speed and simplicity over number of evaluations, meaning it is a "few-time" construction. Depending on configuration this means the key lifetime can be as low as 1 meaning that keys needs to be refreshed after each evaluation.

5.1.3 LaV

LaV [20] is another lattice-based VRF from 2022 that supports a large number of evaluations. Like LB-VRF, it also relies on the MSIS problem alongside MLWR. MLWR is similar to MLWE but uses a deterministic rounding operation instead of noise. LaV further develops standard lattice-based constructions by using a combination of exact and relaxed proofs to reduce overall proof size.

The use of relaxed proofs allows the system to trade off strict, per evaluation verification for efficiency in communication and storage, without compromising correctness. This design makes LaV suitable for constructions with high demands in terms of number of evaluations.

When it was introduced it had the smallest output size among any quantum-safe VRF based on standard assumptions. Compared to other lattice-based VRFs it aims to balance security, proof size and scalability.

5.1.4 X-VRF

X-VRF [30] is based on the XMSS signature scheme, which is a hash-based construction that relies on large Merkle trees for key management. In a Merkle tree, each internal node is the hash of its child nodes, while the leaf nodes contain one-time signature keys.

X-VRF can be instantiated in different configurations depending on the required number of evaluations. Increasing the tree size increases the number of possible evaluations, but also significantly increases tree generation time. A long-lived system may take days to generate, which is a significant drawback from a practical standpoint.

Another limitation was identified in [35], where it was shown that the uniqueness proof can be broken if a hash collision is known, i.e., when two different inputs produce the same hash value. Uniqueness, in this context, refers to the property that each input is associated with exactly one valid VRF output. This property is important because it ensures that the VRF output is unambiguous and deterministic from the perspective of verifiers, preventing the possibility of multiple valid outputs for the same input.

While such attacks are only feasible against weaker or outdated hash functions, the use of modern cryptographic hash functions such as SHA-3 significantly mitigates this risk. In that case, the practical security of the scheme remains intact, even if the formal uniqueness proof does not hold under the same assumptions.

5.1.5 iVRF

Indexed VRF [31] (iVRF) is a recent hash-based VRF construction. It is primarily designed as a post-quantum replacement for ECVRF, which is used in Algorand [17] but is not quantum secure. In Algorand, the VRF plays a central role in leader and committee election.

iVRF achieves post-quantum security by relying entirely on hash functions, avoiding number-theoretic assumptions such as those used in ECVRF. Its construction is based on indexing into a large, structured set of hash outputs, allowing the prover to generate both a pseudorandom output and a corresponding proof of correctness.

The system uses Merkle trees for efficient commitment and verification. This allows a verifier to check that the output was correctly derived from the committed structure without learning any additional information about the secret key. The security of the construction relies primarily on the collision resistance of the underlying hash function, i.e., the assumption that it is computationally infeasible to find two distinct inputs that produce the same hash output.

Since iVRF relies on large Merkle tree structures, the setup phase is computationally expensive. This cost is incurred during initial system setup and whenever the number of allowed evaluations is exhausted.

5.1.6 DeuringVRF

DeuringVRF [21] is based on the $OMIP_{\text{2dim}}$ problem in isogeny-based cryptography. $OMIP_{\text{2dim}}$ stands for the One-More Isogeny Problem in a two-dimensional setting. Informally, the problem captures the difficulty of computing a new isogeny instance given access to an oracle that outputs both the codomain and a two-dimensional isogeny representation. After an unspecified number of oracle queries, an adversary should still have no significant advantage in solving a fresh, previously unqueried instance.

A key characteristic of isogeny-based cryptography is that it relies on relatively new and non-standard hardness assumptions. In contrast to lattice- or hash-based constructions, isogeny-based systems are less well-established and have a shorter cryptanalytic history. Several isogeny-based systems have recently been broken, highlighting the risks associated with such assumptions. This strong dependence on a small set of assumptions is sometimes referred to as brittleness, meaning that even minor weakening of the underlying assumptions may significantly affect security. Ideally, cryptographic constructions rely on multiple, partially independent assumptions to provide layered security, so that weakening one assumption does not immediately break the entire system.

The construction uses Deuring correspondence to simplify the underlying hard problems. Deuring correspondence establishes a connection between supersingular elliptic curves over finite fields and maximal orders in quaternion algebras. This correspondence allows problems about isogenies between elliptic curves to be translated into problems about ideals in quaternion algebras. Such algebraic formulations are

often better understood and can be used to design more structured cryptographic constructions.

5.2 Comparison

When comparing the VRFs listed in Table 5.1, each construction was evaluated with respect to the criteria defined earlier, including quantum security, setup complexity, evaluation speed, key and proof sizes, and whether they satisfy unbiasability.

LB-VRF can be excluded from consideration due to its limited number of evaluations. Although this limitation was not the primary design focus, it provides no compensating advantages that would mitigate this drawback. While LB-VRF satisfies unbiasability, several other constructions achieve the same property while outperforming it across most other relevant dimensions.

LaVs main strengths are the large number of supported evaluations and relatively small output sizes. However, it is never stated whether LaV satisfies unbiasability as some other candidates do. In addition, its key and proof sizes are larger than those of several alternative constructions that satisfy more of the evaluation criteria.

X-VRF provides small key sizes and a configurable number of evaluations depending on the initial setup [30]. However, achieving a high number of evaluations requires significantly increased setup time. No explicit proof or statement regarding unbiasability could be identified in the literature; therefore, it is treated as unknown in this regard. This is a limitation compared to constructions with formally established unbiasability guarantees.

iVRF satisfies all considered criteria. It supports a large number of evaluations and has small key sizes, and it provides a notion of unbiasability under a different formalization [31]. iVRF has a costly initial setup phase.

DeuringVRF also satisfies all considered criteria for a VRF in AC systems. It provides unbiasability, supports a large number of evaluations, and achieves small proof sizes. This makes it a strong candidate for post-quantum applications. However, it relies on isogeny-based assumptions, which are less established compared to lattice- and hash-based constructions.

5.2.1 Comparison of iVRF and DeuringVRF

Among the considered alternatives, iVRF and DeuringVRF stand out as strong candidates for AC systems and fulfill all the requirements.

In terms of efficiency, DeuringVRF generally achieves slightly smaller key and proof sizes compared to iVRF, as well as functionally unbounded evaluations. These properties make it attractive from a performance and scalability perspective. The main advantage of iVRF is its reliance on hash-based constructions, which are based on more established assumptions compared to isogeny-based cryptography. However, iVRF has a notable drawback in terms of setup time. While significantly faster than

X-VRF setup, its initialization cost remains high compared to all other surveyed alternatives.

Regarding the underlying assumptions, hash-based and isogeny-based constructions differ in their maturity and cryptanalytic history. Hash-based constructions are generally considered conservative, as they rely on well-studied primitives whose security has been extensively analysed over time. In contrast, isogeny-based constructions rely on newer and less established hardness assumptions, although recent advances and attacks, such as those on SIDH, do not necessarily imply that all isogeny-based systems are insecure.

At the same time, reliance on well-studied assumptions does not by itself guarantee security, nor does the relative novelty of isogeny-based assumptions imply insecurity. Both approaches therefore represent different trade-offs between maturity and efficiency.

Based on the criteria defined in this work, both iVRF and DeuringVRF satisfy the requirements for AC systems. Consequently, it is not possible to conclusively identify one as strictly superior, as each offers different advantages depending on the prioritised design goals.

5.3 Effect on system function

Having found suitable candidates the next task was to discern if using them would cause any dramatic effect on system function such as epoch duration being much longer. Epoch duration increasing by a factor smaller than 5 might not be hugely problematic, but increases by a factor of 10, 100 or even more would make the VRFs practically unusable as any advantages in effectiveness fade, regardless of system size.

As can be seen in table 5.1 the proof size of iVRF is roughly 8 times larger than for ECVRF. This would mean traffic across any network would be significantly heavier. In the paper proposing iVRF example verification times are listed for both ECVRF and iVRF [31]. The listed verification time and evaluation time of ECVRF are 5 times and 10 times longer respectively than those of iVRF. iVRF has, as earlier mentioned, a setup duration which is high at up to roughly an hour. This means initiation of a system or refreshing keys takes a significant amount of time, something which is not a problem when using ECVRF.

DeuringVRF has a proof size 3 times larger than ECVRF which makes traffic more intense, but not to the point of iVRF. Listed in the original paper [21], there are tests of implementations but on different hardware. As such, no trustworthy conclusions can be drawn from comparing testing times. DeuringVRFs verification and evaluations were above ECVRFs time from the earlier mentioned test.

DeuringVRF is predicted to be slightly slower than ECVRF, judging from proof sizes. As can be seen in iVRF proof sizes are not necessarily an indicator of verification and evaluation times, but indicates the network bandwidth use. For iVRF the setup and key refresh time would slow down most AC systems using it, though only during a rarely occurring event. Proof sizes are also larger, but verification and evaluation

times are shorter. As such it is difficult to accurately predict the effect on epoch duration. Our prediction based on all metrics surveyed is that an increase in epoch duration would likely be relatively minor, and is unlikely to be an increase by a factor of 10 or more.

6

Quorum Order of Growth in Anonymous Committees

The final objective of this thesis is to determine how quorum size grows as a function of the number of nodes in the network. This relationship can be characterized using order of growth, which describes the asymptotic behavior of a function as its input increases.

In distributed systems, order of growth is commonly expressed using Big-O notation. Big-O captures the dominant growth term of a function while ignoring constant factors and lower-order terms. For example, if increasing x by a fixed amount always increases y by a proportional fixed amount, the relationship is linear and has complexity $O(x)$. If the growth rate increases as x becomes larger, the relationship may be quadratic, $O(x^2)$, or even exponential, $O(2^x)$. Conversely, if the growth rate decreases as x increases, the relationship may be logarithmic, $O(\log x)$. Several common growth classes are illustrated in figure 6.1.

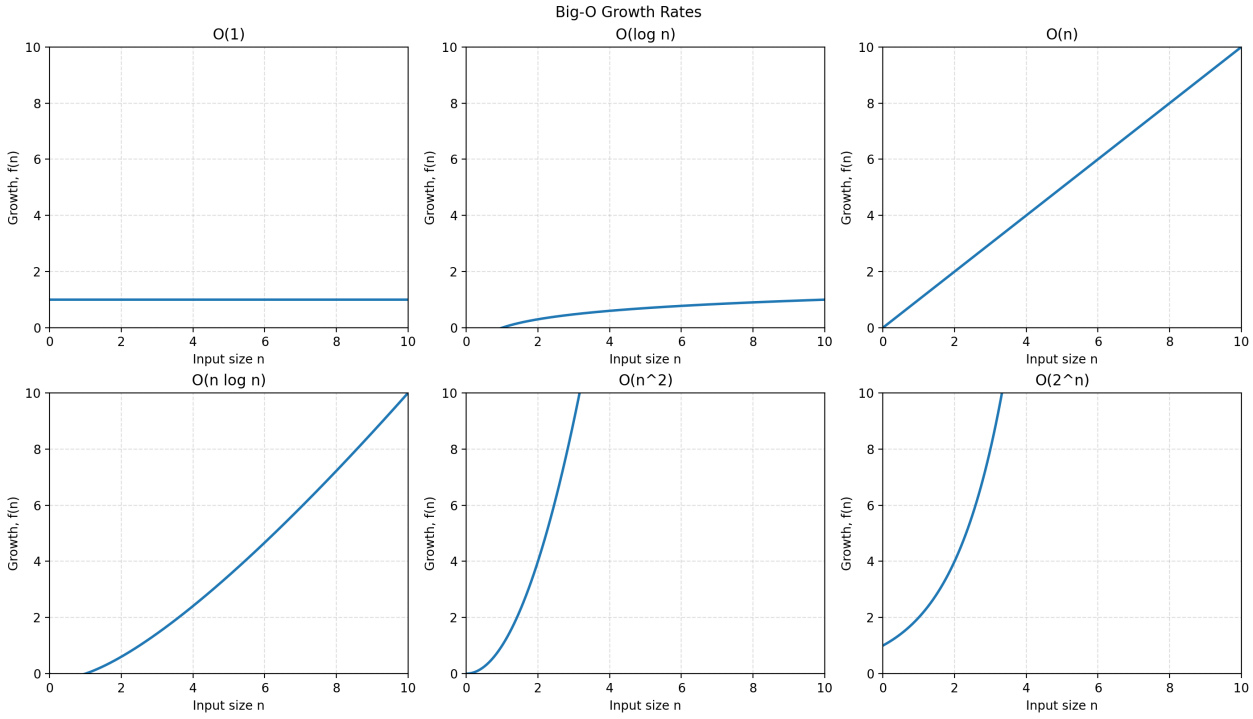


Figure 6.1: Commonly found orders of growth listed in order of growth speed

When using smaller x, order of growth will not have a huge effect, but when x grows into the billions, situations which AC systems are intended to manage, having a linear or exponential order of growth will matter. When calculating these values only the quickest growing factor is deemed relevant. For example $ax^2 + bx + 3c$, where a, b and c are constants has the order of growth $O(x^2)$. This is because ax^2 will be much greater than the other terms for sufficiently large values of x, even if $b \gg a$.

6.1 Approach and hypothesis

The approximation formula for quorums in the "at least one honest party" case in an asynchronous network is $Q \sim pN\sqrt{\frac{(1+f_M)}{2}}$ [6], where f_M is the fraction of malicious nodes, p is the probability of being part of the committee and N is the number of nodes in the system. For this task f_M is not relevant, and $\sqrt{(1+f_M)/2}$ is kept fixed, meaning $Q \sim p \cdot N$. The most relevant question remaining is then how p relates to N while retaining a target bit security. The expectation is that p will decrease, as for example, 1% of 100 nodes is not representative of the entire network, but 1% of a billion nodes might be.

As the code for calculating quorums calculates p using other related numbers, such as target bit security, N, f_M and f_I, we decided to approximate its effect using different combinations of inputs.

6.2 Results

From generating a few values for quorum sizes using 0% inactive nodes, 30% malicious nodes and a target of 128 bit security the result is:

N	Q	p
$5 \cdot 10^4$	3377	0.08318
10^5	3526	0.04341
10^6	3675	0.004523
10^7	3688	0.0004539
10^9	3688	0.000004539
10^{15}	3688	$4.539 \cdot 10^{-12}$
10^{20}	3688	$4.539 \cdot 10^{-17}$

Table 6.1: Number of nodes and associated quorum size. $5 \cdot 10^4$ is present since the simulator stopped working at 10^4.

6.3 Interpretation of results

As can be seen in the table above, the growth from $5 \cdot 10^4$ to 10^5 is relatively quick but from 10^7 to 10^{20} effective quorum size is static. This would indicate some kind of

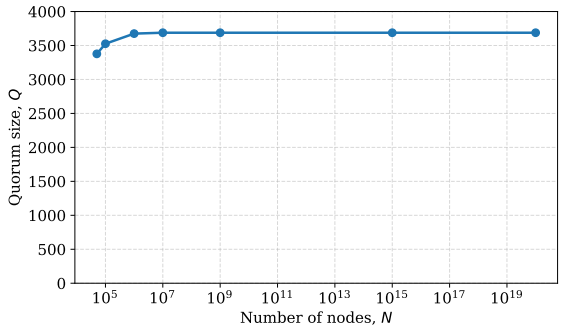


Figure 6.2: Plot of quorum size Q as a function of node count N. X-axis is log-scale.

decreasing growth though decrease is very steep, making it seem as it is approaching an upper bound. N, at 10^7, is increased by a factor of 10^{13} but Q changes by less than $\frac{1}{3688} \approx 0.027\%$. Assuming that the general trend of Q growing slower does not reverse for large N, Q is practically static beyond 10^7. This would mean that, at some point, p decreases at the same rate as N increases. If Q appears effectively constant for large N then the order of growth suggested is consistent with $O(1)$.

After considering the result further a new theory emerged. Simply that a certain amount of nodes will be able to represent the entire system with the granularity necessary. This would mean the order of growth should not be linked to N. In that case, for smaller values of N, it might seem that order of growth is linked to N. However, at larger system sizes it would become increasingly independent. This would mean that in a case where N goes to infinity, quorum size increases becomes independent. This would align well with the explicit goal of AC, to be a more scalable alternative to current methods.

The tendency can be explained by the fact that for lower values of N, the selection pool is finite, which affects the resulting probability distribution. Choosing 25 nodes from a population of 100 where 20 are malicious has a different probability distribution than choosing 25 nodes out of an infinite population where 20% are malicious. For example, choosing more than 20 malicious nodes is impossible if there are only 20 in total, while possible with an infinite population. At the same time, choosing 25 nodes from a population of 10000, where 20% are malicious, has a very similar distribution to the case where N is infinite since the probability that each selected node is malicious is approximately 20%. The trend shown in Table 6.1 reflects the convergence from the finite-population case to the infinite-population limit. For values where $N \gg Q$, Q is independent of changes in N. As the probability distribution where $N = \infty$ is calculable, and results in a quorum which is finite, it implies that quorum size has an upper bound. This means the order of growth implied is $O(1)$. It would also mean that p decreases at the same rate as N grows, meaning $p(N) = \frac{c}{N}$ for some constant c.

Another explanation follows from the fact that the quorum formula is based on a cumulative distribution function (CDF). Consider a fair coin tossed 10 times, which can be modeled as a binomial distribution. The probability density function gives the probability of obtaining each possible number of tails, while the CDF

gives the probability of observing at most a given number of tails. For example, evaluating the CDF at 5 returns the probability of observing at most 5 tails. In general, the CDF approaches 1 as more outcomes are included. Likewise, the quorum formula selects the smallest quorum size Q for which the probability of observing a safe number of malicious nodes exceeds $1 - 2^{-128}$. Once the underlying probability distribution becomes effectively independent of N, the value of Q required to satisfy the probability threshold also becomes effectively independent of N.

Alternatively, implementation details or numerical thresholds in the simulation may contribute to the observed saturation.

6.4 Practical considerations regarding use of VRFs

One practical consideration is the implementation of VRF-based leader or quorum selection. Nodes are typically selected if their VRF output falls below a threshold T, corresponding to a selection probability

$$p = \frac{T}{2^x},$$

where x is the bit length of the VRF output.

In this formulation, the resolution of the VRF is determined by the size of its output space (e.g. 2^{256} for a 32-byte VRF output). This allows for extremely fine-grained probabilities; for example, a 256-bit output supports probabilities as small as

$$2^{-256} \approx 10^{-77},$$

which is far beyond any realistic system requirement.

Table 5.1 includes VRFs up to 1024-bit outputs, corresponding to a theoretical probability resolution of

$$2^{-1024} \approx 10^{-307}.$$

In practice, this implies that VRF precision does not constrain quorum selection for any realistic network size.

A more relevant consideration is not the VRF output size, but how the selection probability p is defined as a function of network size N. If p, f_M were held constant, the expected quorum size would scale linearly as

$$Q \sim N, O(N)$$

However, in systems where p decreases with N, the quorum size may instead saturate, as observed in Table 6.1.

Thus, the observed behavior is unlikely to be limited by VRF precision, but instead reflects the underlying scaling rule governing $p(N)$.

6.5 Comparison in practical use

Relevant to order of growth question is how expensive messaging inside different consensus protocols would be to use.

For comparison, classical distributed consensus protocols such as pBFT [27] exhibit a communication complexity of $O(N^2)$, which becomes prohibitive at large scales. This arises from the all-to-all communication pattern, where each of the N nodes sends messages to approximately $N - 1$ other nodes.

In CoD, which uses AC, the distribution of endorsements constitutes the dominant communication cost. If the quorum size depends on the network size N, and endorsements are disseminated using a star topology, the communication cost is linear in the number of participants in the quorum. This yields a total communication complexity of

$$O(N \cdot Q(N)),$$

where $Q(N)$ denotes the quorum size as a function of N.

If quorum size is asymptotically constant, i.e. $Q(N) = O(1)$ as suggested by the results in Table 6.1, then the total communication complexity reduces to

$$O(N).$$

This is asymptotically more efficient than $O(N^2)$, particularly for large-scale systems where N is very large. In such regimes, the difference in scalability becomes increasingly significant.

7

Conclusion

7.1 Ethical considerations

A common problem with secure end-to-end encryption is that it can be used to coordinate different illegal actions, for example actions of criminal gangs. If the communication would occur over a less secure medium then the chat logs would be evidence in a criminal court. The existence of secure end-to-end communication services has allowed such groups to evade justice, at least for a time [36].

Our view is that while secure communication can be used for malicious purposes, the overall effect is positive. In an oppressive society the users might not be criminals in our view. Surveillance might be commonplace making normal means of contact dangerous, and then secure communication might be the only option for expressing freedom of thought.

As such privacy is overall deemed to be a noble aspiration, with some unfortunate side effects.

7.2 Summary of findings

In this thesis we investigated the robustness, quantum safety and scalability of ACs. We have found that incorrect assumptions of malicious and inactive nodes affect security in unintuitive ways, and that this depends a lot on the kind of AC.

After formulating a methodology for finding suitable post-quantum VRFs two strong candidates were found. According to benchmarks they should have no adverse effects on system function in CoD or general AC protocols.

The order of growth dependent on node count has been found to be consistent with $O(1)$, as quorum size seems to approach an upper bound for large node counts.

Bibliography

[1] "Consistency-or-Die: Consistency in Star Topologies from Dynamic Committees applied to Key Transparency," Private communication, more recent version of https://eprint.iacr.org/2024/879.

[2] J. Amann, O. Gasser, Q. Scheitle, L. Brent, G. Carle, and R. Holz, "Mission accomplished? HTTPS security after diginotar," in *Proceedings of the 2017 Internet Measurement Conference*, ser. IMC '17, New York, NY, USA: Association for Computing Machinery, Nov. 1, 2017, pp. 325–340, ISBN: 978-1-4503-5118-8. DOI: 10.1145/3131365.3131401. Accessed: Jun. 29, 2026. [Online]. Available: https://dl.acm.org/doi/10.1145/3131365.3131401.

[3] L. Chuat, P. Szalachowski, A. Perrig, B. Laurie, and E. Messeri, "Efficient gossip protocols for verifying the consistency of certificate logs," in *2015 IEEE Conference on Communications and Network Security (CNS)*, 2015, pp. 415–423. DOI: 10.1109/CNS.2015.7346853.

[4] V. Buterin, *Ethereum: A next-generation smart contract and decentralized application platform*, https://ethereum.org/en/whitepaper/, Accessed: 2026-05-08, 2014.

[5] J. Bonneau, "EthIKS: Using Ethereum to Audit a CONIKS Key Transparency Log," in *Financial Cryptography and Data Security*, J. Clark, S. Meiklejohn, P. Y. Ryan, D. Wallach, M. Brenner, and K. Rohloff, Eds., Berlin, Heidelberg: Springer, 2016, pp. 95–105, ISBN: 978-3-662-53357-4. DOI: 10.1007/978-3-662-53357-4_7.

[6] B. David, L. Lavagnino, E. Pagnin, and P. Stankovski Wagner, "Tail-Hammer: Optimized statistics for anonymous committees and applications," in *Proceedings of the 15th International Conference on Security and Cryptography for Networks (SCN 2026)*, To appear, Springer, 2026.

[7] E. Blum, D. Leung, J. Loss, J. Katz, and T. Rabin, "Analyzing the Real-World Security of the Algorand Blockchain," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '23, New York, NY, USA: Association for Computing Machinery, Nov. 21, 2023, pp. 830–844, ISBN: 979-8-4007-0050-7. DOI: 10.1145/3576915.3623167. Accessed: Feb. 10, 2026. [Online]. Available: https://dl.acm.org/doi/10.1145/3576915.3623167.

[8] J. Brorsson, E. Pagnin, B. David, and P. S. Wagner. "Consistency-or-Die: Consistency for Key Transparency," Accessed: Jan. 26, 2026. [Online]. Available: https://eprint.iacr.org/2024/879, pre-published.

[9] R. Campbell, "Enterprise Migration to Post-Quantum Cryptography: Timeline Analysis and Strategic Frameworks," *Computers*, vol. 15, no. 1, p. 9, Jan. 2026, ISSN: 2073-431X. DOI: 10.3390/computers15010009. [Online]. Available: https://www.mdpi.com/2073-431X/15/1/9.

[10] P. W. Shor, "Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer," *SIAM Journal on Computing*, vol. 26, no. 5, pp. 1484–1509, 1997. DOI: 10.1137/S0097539795293172. eprint: https://doi.org/10.1137/S0097539795293172. [Online]. Available: https://doi.org/10.1137/S0097539795293172.

[11] P. C. Florindo, "Realizing consistency-or-die: Verifiable consistency for key logs," M.S. thesis, Department of Computer Science and Engineering, Chalmers University of Technology, 2025.

[12] C. Cachin, R. Guerraoui, and L. Rodrigues, *Introduction to Reliable and Secure Distributed Programming*, 2nd. Springer, 2011.

[13] W. Vogels, "Eventually consistent," *Commun. ACM*, vol. 52, no. 1, pp. 40–44, Jan. 2009, ISSN: 0001-0782. DOI: 10.1145/1435417.1435432. [Online]. Available: https://doi.org/10.1145/1435417.1435432.

[14] L. Lamport, R. E. Shostak, and M. C. Pease, "The byzantine generals problem," *ACM Transactions on Programming Languages and Systems*, vol. 4, no. 3, pp. 382–401, 1982. DOI: 10.1145/357172.357176. [Online]. Available: https://lamport.azurewebsites.net/pubs/byz.pdf.

[15] M. F. Esgin et al., "Practical Post-quantum Few-Time Verifiable Random Function with Applications to Algorand," in *Financial Cryptography and Data Security*, N. Borisov and C. Diaz, Eds., Berlin, Heidelberg: Springer, 2021, pp. 560–578. DOI: 10.1007/978-3-662-64331-0_29.

[16] Y. Dodis and A. Yampolskiy, "A Verifiable Random Function with Short Proofs and Keys," in *Public Key Cryptography - PKC 2005*, S. Vaudenay, Ed., Berlin, Heidelberg: Springer, 2005, pp. 416–431, ISBN: 978-3-540-30580-4. DOI: 10.1007/978-3-540-30580-4_28.

[17] Y. Gilad, R. Hemo, S. Micali, G. Vlachos, and N. Zeldovich, "Algorand: Scaling Byzantine Agreements for Cryptocurrencies," in *Proceedings of the 26th Symposium on Operating Systems Principles*, ser. SOSP '17, New York, NY, USA: Association for Computing Machinery, Oct. 14, 2017, pp. 51–68, ISBN: 978-1-4503-5085-3. DOI: 10.1145/3132747.3132757. Accessed: Jan. 26, 2026. [Online]. Available: https://dl.acm.org/doi/10.1145/3132747.3132757.

[18] B. David, P. Gai, A. Kiayias, and A. Russell, "Ouroboros Praos: An Adaptively-Secure, Semi-synchronous Proof-of-Stake Blockchain," in *Advances in Cryptology EUROCRYPT 2018*, J. B. Nielsen and V. Rijmen, Eds., Cham: Springer International Publishing, 2018, pp. 66–98, ISBN: 978-3-319-78375-8. DOI: 10.1007/978-3-319-78375-8_3.

[19] J. Wohlwend, *Elliptic curve cryptography: Pre and post quantum*, https://math.mit.edu/~apost/courses/18.204-2016/18.204_Jeremy_Wohlwend_final_paper.pdf, Accessed: 2025-12-12, 2016.

[20] M. F. Esgin, R. Steinfeld, D. Liu, and S. Ruj, "Efficient Hybrid Exact/Relaxed Lattice Proofs and Applications to Rounding and VRFs," in *Advances in Cryptology CRYPTO 2023*, H. Handschuh and A. Lysyanskaya, Eds., Cham:

Springer Nature Switzerland, 2023, pp. 484–517. DOI: 10.1007/978-3-031-38554-4_16.

[21] A. Leroux, "Verifiable Random Function from the Deuring Correspondence and Higher Dimensional Isogenies," in *Advances in Cryptology EUROCRYPT 2025*, S. Fehr and P.-A. Fouque, Eds., Cham: Springer Nature Switzerland, 2025, pp. 167–194, ISBN: 978-3-031-91098-2. DOI: 10.1007/978-3-031-91098-2_7.

[22] W. Castryck and T. Decru, *An efficient key recovery attack on SIDH*, Cryptology ePrint Archive, Paper 2022/975, 2022. [Online]. Available: https://eprint.iacr.org/2022/975.

[23] B. Laurie, E. Messeri, and R. Stradling, *Certificate Transparency Version 2.0*, RFC 9162, Dec. 2021. DOI: 10.17487/RFC9162. [Online]. Available: https://www.rfc-editor.org/info/rfc9162.

[24] C. Gentry et al., *YOSO: You only speak once / secure MPC with stateless ephemeral roles*, Cryptology ePrint Archive, Paper 2021/210, 2021. [Online]. Available: https://eprint.iacr.org/2021/210.

[25] M. S. Melara, A. Blankstein, J. Bonneau, E. W. Felten, and M. J. Freedman, "CONIKS: Bringing Key Transparency to End Users," presented at the 24th USENIX Security Symposium (USENIX Security 15), 2015, pp. 383–398, ISBN: 978-1-939133-11-3. Accessed: Apr. 10, 2026. [Online]. Available: https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/melara.

[26] J. Len, M. Chase, E. Ghosh, K. Laine, and R. C. Moreno, *OPTIKS: An optimized key transparency system*, Cryptology ePrint Archive, Paper 2023/1515, 2023. [Online]. Available: https://eprint.iacr.org/2023/1515.

[27] M. Castro and B. Liskov, "Practical byzantine fault tolerance," in *Proceedings of the Third Symposium on Operating Systems Design and Implementation*, ser. OSDI '99, New Orleans, Louisiana, USA: USENIX Association, 1999, pp. 173–186, ISBN: 1880446391.

[28] Anonymous, https://anonymous.4open.science/r/tail-hammer-F253/README.md, version uploaded on Feb 4th, 2026.

[29] https://github.com/st0ick/Master-Thesis-Consensus-via-Anonymous-Committees, version uploaded on Jun 27th, 2026.

[30] M. Buser et al., "Post-Quantum Verifiable Random Function from Symmetric Primitives in PoS Blockchain," in *Computer Security ESORICS 2022*, V. Atluri, R. Di Pietro, C. D. Jensen, and W. Meng, Eds., Cham: Springer International Publishing, 2022, pp. 25–45, ISBN: 978-3-031-17140-6. DOI: 10.1007/978-3-031-17140-6_2.

[31] M. F. Esgin et al., "A New Look at Blockchain Leader Election: Simple, Efficient, Sustainable and Post-Quantum," in *Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security*, ser. ASIA CCS '23, New York, NY, USA: Association for Computing Machinery, Jul. 10, 2023, pp. 623–637, ISBN: 979-8-4007-0098-9. DOI: 10.1145/3579856.3595792. Accessed: Feb. 25, 2026. [Online]. Available: https://dl.acm.org/doi/10.1145/3579856.3595792.

[32] D. Papadopoulos et al. “Making NSEC5 Practical for DNSSEC,” Accessed: Apr. 16, 2026. [Online]. Available: https://eprint.iacr.org/2017/099, pre-published.

[33] A. Langlois and D. Stehle, *Worst-case to average-case reductions for module lattices*, Cryptology ePrint Archive, Paper 2012/090, 2012. [Online]. Available: https://eprint.iacr.org/2012/090.

[34] Y. Li, S. Liu, S. Han, D. Gu, and J. Weng, “Simulatable verifiable random function from the LWE assumption,” *Theoretical Computer Science*, vol. 957, p. 113 826, May 12, 2023, ISSN: 0304-3975. DOI: 10.1016/j.tcs.2023.113826. Accessed: Feb. 24, 2026. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0304397523001391.

[35] O. Bodaghi and R. Safavi-Naini, “Short paper: Breaking x-vrf, a post-quantum verifiable random function,” in *Financial Cryptography and Data Security: 28th International Conference, FC 2024, Willemstad, Curaçao, March 48, 2024, Revised Selected Papers, Part II*, Willemstad, Curaçao: Springer-Verlag, 2024, pp. 90–100, ISBN: 978-3-031-78678-5. DOI: 10.1007/978-3-031-78679-2_5. [Online]. Available: https://doi.org/10.1007/978-3-031-78679-2_5.

[36] D. Goodin, “Police decrypt 258,000 messages after breaking pricey ironchat crypto app,” *Ars Technica*, Nov. 2018. [Online]. Available: https://arstechnica.com/information-technology/2018/11/police-decrypt-258000-messages-after-breaking-pricey-ironchat-crypto-app/.