Protocols and Lower Bounds for Failure Localization in the Internet

Boaz Barak Sharon Goldberg David Xiao
Princeton University, Princeton, NJ 08544

This is the full version of [6] from July 20, 2009.

Abstract

A secure *failure-localization path-quality-monitoring* (FL-PQM) protocol allows a sender to localize faulty links on a single path through a network to a receiver, even when intermediate nodes on the path behave adversarially. Such protocols were proposed as tools that enable Internet service providers to select high-performance paths through the Internet, or to enforce contractual obligations. We give the first formal definitions of security for FL-PQM protocols and construct:

1. A simple FL-PQM protocol that can localize a faulty link *every time* a packet is not correctly delivered. This protocol's communication and storage overhead is $O(1)$ additional messages of length $O(n)$ per packet (where n is the security parameter).
2. A two more efficient FL-PQM protocols that can localize a faulty link when a *noticeable fraction* of the packets sent during some time period are not correctly delivered. Our sampling-based protocol has a storage and communication overhead that is an arbitrarily small fraction of the total number of packets sent T. Our sketching-based protocol requires $O(n + \log T)$ storage and only two additional messages of similar length.

We also prove lower bounds for such protocols:

1. Every secure FL-PQM protocol requires *each* intermediate node on the path to have some shared secret information (*e.g.,* keys).
2. If secure FL-PQM protocol exist then so do one-way functions.
3. Every *black-box* construction of a FL-PQM protocol from a random oracle that securely localizes every packet and adds at most $O(\log n)$ messages overhead per packet requires *each* intermediate node to invoke the oracle.

These results show that implementing FL-PQM requires active cooperation (*i.e.,* maintaining keys and agreeing on, and performing, cryptographic protocols) from *all* of the intermediate nodes along the path. This may be problematic in the Internet, where links operate at extremely high speeds, and intermediate nodes are owned by competing business entities with little incentive to cooperate.

Keywords. Failure localization, secure routing, black-box separation.

1 Introduction

The Internet is an indispensable part of our society, and yet its basic foundations remain vulnerable to attack. Secure routing protocols seek to remedy this by not only providing guarantees on the correct setup of paths from sender to receiver through a network (*e.g.,* Secure BGP [21]), but also by verifying that data packets are actually delivered correctly along these paths. Packet delivery is surprisingly susceptible to simple attacks; in the current Internet, packets are typically sent along a single path from sender to receiver, and so a malicious node along the data path can easily drop or modify packets before they reach their destination. While small amounts

of random packet loss are considered to be a natural part of the Internet's operation, there are many situations in which a sender would like to detect and respond to unusually high rates of packet loss or corruption along a path. To this end, the networking community has recently been studying monitoring and measurement protocols that return information about packet loss events on a data path (*e.g.,* [9, 11, 23, 28, 27, 5, 4, 25, 3]). The motivation for these protocols is twofold. First, they provide the sender with information that he can use during path setup to select a single, high-performance path to the receiver from the multiple available paths through the network [17]. Second, since Internet service is a contractual business, where senders pay nodes along the data path to carry their packets, information from Internet measurement protocols is highly valuable for enforcing contractual obligations between nodes. Indeed, a number of works [22,8,3] argue that quality of service on the Internet will could degrade unacceptably that if there is a lack of *accountability*, *i.e.,* mechanisms that empower senders to detect and respond to degraded performance on a data path that violate contractual obligations. Note also that if Internet measurement protocols are used to enforce contracts, nodes may have an economic incentive to bias the information obtained from these protocols.

In this work we provide a rigorous cryptographic examination of *secure* monitoring protocols that are robust even in the presence of malicious nodes on the data path. In particular, we study techniques that allow a sender to *localize* the specific links along the data path where packets were dropped or modified— a task that we call *failure-localization path-quality monitoring*. While some protocols for this task are deployed in the Internet today (*e.g.,* traceroute [1]), they are not robust to nodes that behave adversarially in order to bias measurements.

1.1 Our results

We make the following contributions to the study of secure failure-localization path-quality monitoring protocols (in the rest of the paper we call these simply *failure localization* or FL protocols). Throughout the paper, we use the word "packet" to denote data that the sender wishes to transmit, and "message" to refer to both data packets and FL-protocol-related control messages.

Definition. In Section 2, we give the first formal definition of security for failure localization protocols. We note that some of the previous FL protocols suggested in the literature, such as [27, 5, 3], do *not* satisfy our definition. (We sketch attacks in Appendix A.)

We give two variants of the definition— *per-packet* security requires localizing a link each time a packet is not delivered, while *statistical* security only requires this when a noticeable fraction of packets fail to arrive. An important feature of our definition is that it accounts for the fact that messages can be dropped in the Internet for benign reasons like congestion. We note that care must be taken to design protocols that are simultaneously robust to both adversarial behavior and benign congestion. We discuss the effect of this assumption on some previous work [5] in Appendix A.

Protocols. We present three simple protocols satisfying our per-packet (Section 3.1) and statistical (Section 3.2) security definitions. All of these protocols do not modify the packets sent on the path; instead, they add additional messages. Thus our protocols have the important advantage of allowing backwards compatibility with the current techniques for processing packets in a router, minimizing latency in the router, and not increasing packet size.

Because routers are highly-resource constrained devices that are designed to communicate large amounts of information while storing very little, the most important measure of efficiency for our protocols is storage overhead (*i.e.,* the amount of state each router needs to keep as part of the protocol). We are also concerned with communication overhead (*i.e.,* the number and size of messages added by the protocols), and the computational overhead (*i.e.,* the complexity of the computation that each router needs to perform per packet that it processes).

Our per-packet protocol requires each router to store an $O(n)$-length tag for each packet

that it sends, and adds a single $O(n)$-length message to every packet sent (n is the security parameter), and one $O(Kn)$-length messages when a failure occurs. (Typically in the Internet, the path length K is less than 20, when nodes represent individual routers, and when nodes represent Internet Service Providers (ISPs) then there are on average $K \approx 4$, and no more 10 nodes on a typical path [21].) However, the communication and storage overhead of the protocol is considered severe, so we present this mostly for pedagogical purposes and move on to our statistical FL protocols.

Our first statistical protocol is based on sampling, and needs to store and communicate $O(pT)$ tags of length $O(n)$ each when the sampling rate is p and T packets are sent. For clarity and correctness, we present a version of the protocol based on the Symmetric Secure Sampling protocol from [13, 12] ; this version of the protocol requires each intermediate node to share *symmetric cryptographic keys* with Alice *and* Bob. However, we emphasize that is possible to construct an analogous statistical PQM protocol for the public-key setting as well, based on the Asymmetric Secure Sampling protocols in [13, 12] . Such a protocol would require each node to perform a similar amount of symmetric cryptographic operations on a per-packet basis, and require only a single public-key cryptographic operation for each T packets sent.

Next, we present much more efficient statistical FL protocol based on the *secure sketching* protocol from [13, 12] . This protocol requires the each node to share a symmetric key with Alice only, and requires each node to store a single $O(K^2 \log T)$-sized array of counters, called a sketch. The communication overhead of the protocol is only two additional messages of length $O(K^3 \log T + Kn)$ for every T packets sent, and we do not require any modifications to the packets sent by Alice. However, unlike our sampling-based protocols, this protocol cannot be generalized to public-key setting.

Lower bounds. Like many of the protocols in the literature [4, 5, 27, 30, 25, 3], both of our protocols require cryptographic keys and computations at each node. These requirements are considered severe in the networking literature; setting up a key infrastructure and agreeing on cryptographic primitives is challenging in the distributed world of the Internet, where each node is owned by a different entity with sometimes incompatible incentives. However, in Section 4 we show that these requirements are to some degree *inherent* by:

1. Proving that every secure (per-packet or statistical) FL protocol requires a key infrastructure, or more precisely, that intermediate nodes and Alice and Bob must all share some secret information between each other. This shared secret information can be pairwise symmetric keys, or public-private key pairs.

2. Proving that a one-way function can be constructed from any secure FL protocol.

3. Giving evidence that any practical per-packet secure FL protocol must use these keys in a cryptographic way at *every node* (*e.g.,* , it does not suffice to use the secret information with some simple, non-cryptographic, hash functions as in [11]). We show that in every black-box construction of such a protocol from a random oracle, where at most $O(\log n)$ protocol messages are added per packet, then every intermediate node must query the random oracle. We note that practical protocols designed for Internet routers typically avoid using non-black-box constructions or adding more than a constant number of protocol messages per packet. We also show that for statistically-secure FL, or FL protocols adding $\omega(\log n)$ messages per packet, the necessity of cryptography depends on subtle variations in the security definition.

Implications of our results. Our lower bounds raise questions about the practicality of deploying FL protocols. In small highly-secure networks or for certain classes of traffic, the high key-management and cryptographic overhead required for FL protocols may be tolerable. However, FL protocols may be impractical for widespread deployment in the Internet; firstly

because intermediate nodes are owned by competing business entities that may have little incentive to set up a key infrastructure and agree on cryptographic protocols, and secondly because cryptographic computations are expensive in the core of the Internet, where packets must be processed at extremely high speeds (about 2 ns per packet). Thus, our work can be seen as a motivation for finding security functionalities for the Internet that are more practical than failure localization.

1.2 Related work

Some of this work (in particular, the results of Section 3 and a weaker version of Theorem 4.3) appeared in our earlier technical report [12]. We built on [12] in [13], where, together with Jennifer Rexford and Eran Tromer, we gave formal definitions, constructions, and lower bounds for the simpler task of *path-quality monitoring* (PQM). In a PQM protocol the sender only wishes to *detect* if a failure occurred, rather than localize the specific faulty link along the path. We use the results from [13, 12] in Section 3.2 to show how a PQM protocol can be composed to obtain a statistical FL protocol, and in Section 4.2 to argue that FL protocols need cryptographic computations.

In addition to the FL protocols from the networking literature [4, 5, 27, 25, 3, 29], our work is also related to the work on secure message transmission (SMT) begun by Dolev, Dwork, Waart, and Yung in [10]. In SMT, a sender and receiver are connected by a multiple parallel wires, any of which can be corrupted by an adversary. Here, we consider a single path with a series of nodes that can be corrupted by an adversary, instead of multiple parallel paths. Furthermore, while multiple parallel paths allow SMT protocols to *prevent* failures, in our single path setting, an adversarial intermediate node can always block the communication between sender and receiver. As such, here we only consider techniques for *detecting and localizing* failures.

Subsequent to the publication of this work in [6], Zhang *et al.* [30] considered FL protocols that are similar to our per-packet FL and our sampling-based statistical FL protocols. Furthermore, Amir, Bunn, and Ostrovsky [2] consider FL in the setting of multiple paths, as in the SMT framework.

2 Our model

In a failure localization (FL) protocol, a sender Alice wants to know whether the packets she sends to receiver Bob arrive unmodified, and if not, to find the link along the path where the failure occurred (see Figure 1). We say a *failure* or *fault* occurs when a data packet that was sent by Alice fails to arrive unmodified at Bob. Following the literature, we make the somewhat strong assumption that Alice knows the identities of all the nodes of the data path. While this assumption only strengthens our lower bounds, it does limit the practicality of our protocols in settings where Alice is not sure about the paths her packets take. For more discussion on this assumption, see [**?**]. We work in the setting where all traffic travels on symmetric paths (*i.e.,* intermediate nodes have bi-directional communication links with their neighbors, and messages that sender Alice sends to receiver Bob traverse the same path as the messages that Bob sends back to Alice). We say that messages travelling towards Alice are going *upstream*, and messages travelling towards Bob are going *downstream*. An adversary Eve can occupy any set of nodes on the path between Alice and Bob, and can add, drop, or modify messages sent on the links adjacent to any of the nodes she controls. She can also use timing information to attack the protocol.

Localizing links, not nodes. It is well known that an FL protocol can only pinpoint a *link* where a failure occurred, rather than the *node* responsible for the failure. To see why, refer to Figure 1, and suppose that (a) Eve controlling node R_2 becomes unresponsive by ignoring all the messages she receives from R_1. Now suppose that (b) Eve controls node R_1 and pretends

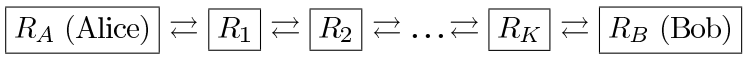

Figure 1: A path from Alice to Bob via K intermediate nodes.

that R_2 is unresponsive by dropping all communication to and from R_2. Because cases (a) and (b) are completely indistinguishable from Alice's point of view, at best Alice can localize the failure to link $(1, 2)$.

Congestion. Congestion-related packet loss is widespread on the current Internet, caused by protocols like TCP [20] that naturally drive the network into a state of congestion. Our definition accounts for congestion by assuming links can drop each message independently with some probability. One could come up with other models for congestion (*e.g.,* allowing Eve to specify the distribution of congestion-related packet loss), and for some plausible choices our positive results will still hold. However, we use independent drops for the sake of simplicity. Furthermore, assuming that congestion is not controlled by the adversary only strengthens our lower bounds and makes our model more realistic.

2.1 Security definition

Let n be the security parameter. A failure localization protocol consists of an efficient initialization algorithm Init taking n uniformly random bits and generating keys for each node, and efficient node algorithms Alice, Bob, $R_1, \ldots, R_K$ which take in a key and communicate with each other as in Figure 1. We always fix $K = O(1)$ independent of n. The Alice algorithm takes in a packet that she wants to send to Bob. If communication is successful, then the Bob algorithm outputs the packet that Alice sent. Our security definitions are game-based:

Definition 2.1 (Security game for FL)**.** *The game begins when Eve chooses a subset of nodes $E \subseteq \{1, \ldots, K\}$ that she will occupy for the duration of the game. The Init algorithm is then used to generate keys for each node, and Eve is given the keys for the nodes $i \in E$ that she controls. We define an oracle Source that generates data packets d for the Alice algorithm to send. We allow Eve to choose the packets that the Source oracle generates, subject to the condition that she may not choose the same packet more than once during the game.*[1]

We allow Eve to add, drop, or modify any of the messages sent on the links adjacent to the nodes she occupies. We include congestion in our model by requiring that, for each message sent on each link on the path, the link goes down *or drops the message with some constant probability $\rho > 0$. Notice that this means that a failure can happen at links not adjacent to a node occupied by Eve.*

We introduce the notion of time into our model by assuming that the game proceeds in discrete time steps; in each time step, a node can take in an input and produce an output, and each link can transmit a single message. (Thus, each time step represents an event occurring on the network.) Because it is expensive to have securely synchronized clocks in a distributed system like the Internet,[2] *we do* not *allow the honest algorithms to take timing information as an input. However, to model timing attacks, we assume that Eve knows which time step that the game is in.*

Then, our per-packet security definition uses the game defined in Definition 2.1:

[1]We make this assumption because there is natural entropy in packet contents, due to TCP sequence numbers and IP ID fields [11]. One way to enforce this assumption in practice, is to require the use of ephemeral 'interval keys' which are refreshed at the end an interval, and to assume that the natural entropy in packet contents enforces the uniqueness of packets over an interval (see the further discussion in [13, 12]).

[2]Indeed, the NTP protocol used for clock synchronization on the Internet is not secure [24], and thus should not be used as an input to a secure FL protocol.

Definition 2.2 (Per-packet security for FL)**.** *In the per-packet security game, Eve gets to interact with the* Source *oracle and the "honest" node algorithms as in Definition 2.1, until she decides to stop. For each packet sent, Alice must output either* $\surd$ *(*i.e.*, not raise an alarm) or a link* ℓ *(*i.e.*, raise an alarm and localize a failure to* ℓ*). We assume that the game is* sequential*: Alice must output a decision for each data packet before starting to transmit the next data packet (see remarks below). We say that an FL protocol is* per-packet secure *if the following hold:*

1. *(*Secure localization*). For every packet d sent by the* Source *oracle that is not successfully output by Bob, then Alice outputs a link* ℓ *such that either (a) link* ℓ *is adjacent to a node occupied by Eve, or (b) link* ℓ *went down due to congestion for one of the messages (including FL protocol messages) associated with sending packet d from Alice to Bob.*

2. *(*No false positives*). For every packet d sent by the* Source *oracle that is successfully output by Bob, for which there was no congestion, and for which Eve does not deviate from the protocol, Alice outputs* $\surd$*.*

We need to introduce a few new concepts for our statistical security definition. First, we define an *interval* as a sequence of T packets (and associated FL protocol messages) that Alice sends to Bob.[3] Next, we use the following parameters: a false alarm threshold α, a detection threshold for the path β (where $0 < \alpha < \beta < 1$) and an error parameter $\delta \in \{0, 1\}$. Usually, we will set α such that congestion alone almost never causes the failure rate on a path to exceed the false alarm threshold.

Definition 2.3 ((α, β, δ)-Statistical security for FL)**.** *In the statistical security game, Eve is allowed to choose the number of* intervals *for which she wants to interact with the* Source *oracle and the honest nodes as in Definition 2.1. The number of packets per interval T may grow with n, but is always at least some minimum number depending* α, β, δ, K*. At the end of each* interval*, Alice needs to output either* $\surd$ *(*i.e.*, not raise an alarm) or a link* ℓ *(*i.e.*, raise an alarm and localize a link). The game is sequential; Alice must output a decision for each interval before starting the next interval. Then, an FL protocol is* statistically secure *if the following hold:*

1. *(*Secure localization*). For any interval in the security game where Eve causes the failure rate on the path to exceed the detection threshold* β*, then with probability* $1 - \delta$ *Alice raises alarm for a link* ℓ *that is adjacent to Eve, or a link* ℓ *whose failure rate exceeds* $\frac{\alpha}{K+1}$*.*

2. *(*Few false positives*). For any interval in the security game where Eve does not deviate from the correct algorithm* R_i *of any of the nodes* $i \in E$ *that she controls and the failure rate on each link is below the (per-link) false alarm threshold* $\frac{\alpha}{K+1}$*, then the probability that Alice outputs* $\surd$ *is at least* $1 - \delta$*.*

We now discuss some properties of our security definition.

Benign and malicious failures. Our security definitions require Alice to accurately localize failures, but these failures may be caused by Eve, or may be the result of *benign causes*, such as congestion. We do not require Alice to distinguish between benign or malicious (*i.e.,* due to Eve) failures, because Eve can always drop packets in a way that "looks like" congestion.

Sequential games. For simplicity, in our per-packet security game we required Alice to make FL decisions before she sends a new data packet. This is to capture the fact that such protocols should provide "real-time" information about the quality of the paths she uses, and so we did not allow Alice in the per-packet case to make decisions only after sending many packets (as is done in the statistical security case). We note that while our lower bounds (*i.e.,* attacks) are sequential, our positive results (*i.e.,* , protocols) do not use the assumption of sequential

[3] We can think of an interval as all the packets sent in some time period (*e.g.,* approximately 10^7 packets are sent 100 msec over a 5 Gbps Internet path).

execution in any way, and are secure in a more general setting where Eve can choose, at each point in time, which of the previously sent packets "time-out", and then Alice needs to output FL decisions for these packets. We emphasize that the sequential assumption does *not* prevent Alice from keeping state and using information from *past* packets in order to make FL decisions. (Though none of our positive results require that Alice does this.)

Movements of the adversary. Our model does not allow Eve to move from node to node in a single security game. This assumption, which only strengthens our lower bounds, does not significantly limit the practicality of our protocols for a number of reasons. Firstly, when Eve models a Internet service provider that tries to bias the results of FL protocol for business reasons, it is reasonable to assume that she may only occupy nodes owned by her business entity. Furthermore, when Eve is an external attacker or virus that compromises a router, "leaving" a router means that the legitimate owner of the router removed the attacker from the router, *e.g.,* by refreshing its keys. We model this key refresh process as a re-start of the security game. Furthermore, in practice "movements" to a new router happen infrequently, since an external attacker typically needs a different strategy each time it compromises a router owned by a different business entity.

Generalizations. All our results generalize to the setting where congestion rates, false alarm thresholds, and detection thresholds are different per link; we set them all equal here for simplicity. Our lower bounds also hold for the weaker adversary model where Eve can occupy only one node and the Source oracle generates independent (efficiently-samplable) packets from a distribution that is *not* controlled by Eve.

3 Protocols

We now present protocols for secure per-packet and statistical FL. Our protocols are related, though not identical to those of [3,4,5]. (In Appendix A we show that the protocols in [3,5] do not satisfy our security definitions.)

We use the notation $[m]_k$ to denote a message m authenticated by a key k using a *message authentication code* (MAC); such schemes can be constructed from any one-way function [15,16]. We'll often use the well-known notion of an *onion report*: if every node R_i wants to transmit a report τ_i to Alice in an authenticated way, then we define inductively $\theta_{K+1} = [(K+1, \tau_{\mathsf{Bob}})]_{k_{\mathsf{Bob}}}$ and for $1 \leq i \leq K$, $\theta_i = [(i, \tau_i, \theta_{i+1})]_{k_i}$. That is, each R_i's report is appended with its downstream neighbors' reports before being authenticated and passed upstream. Onion reports prevent Eve from selectively dropping reports — if Eve occupies R_j and wants to drop the report τ_j of R_i for some $i > j$ then, under the assumption that Eve cannot forge MACs, Alice will discover that R_j tampered with the onion report. We also note that every time we send or store a packet d in acknowledgments and reports, we could save space by replacing d with an $O(n)$-length hash of d via some collision-resistant hash function, where n is the security parameter.

3.1 Optimistic Per-Packet FL Protocol

We assume that each node R_i shares a symmetric key k_i with Alice. For each packet that Alice sends, the protocol proceeds in two phases:

The detect phase. Alice stores each packet d that she sends to Bob. When Bob receives the packet d, he responds with an ack of the form $a = [d]_{k_B}$. Alice removes the packet d from storage when she receives a validly MAC'd corresponding ack, and raises an alarm if a valid ack is not received.[4] We also require each intermediate node to store each data packet and corresponding ack.

[4] In practice, each packet d should be stored along with a local timeout at Alice. If the ack does not arrive before the timeout expires, then Alice should raise an alarm.

The localize phase. This phase is run only if Alice raises an alarm for a packet d. Alice sends an *onion report request* $q = (\mathsf{report}, d)$ downstream towards Bob. To respond to the request, each node R_i checks if he stored data packet d; if he did, R_i sets $\tau_i = (q, i, d, a)$ where a is the ack he saw corresponding to packet d, and substituting the symbol $\perp$ for d and/or a if he failed to receive that packet or an ack. R_i then creates an onion report θ_i using τ_i as described above. In the onion report, R_i can substitute the symbol $\theta_{i+1} = \perp$ if he fails to receive a θ_{i+1} from R_{i+1}. [5]

To localize the failure, Alice classifies the onion reports that she received in response to her onion report request q. An onion report $\theta_i = [q', i', d', a', \theta_{i+1}]_{k_i}$ is "consistent" if it is present, *i.e.*, $\theta_i \neq \perp$, and all of the following four conditions hold. Otherwise, an onion report is "inconsistent".

1. $q' = q$ sent out by Alice.
2. The MAC on θ_i is valid.
3. $d' = d$, where d is the packet queried in q.
4. a' is *not* a valid ack for packet d.

Alice localizes then localizes the upstream-most link $(i, i+1)$ where the onion reports transition from consistent to inconsistent.

Theorem 3.1. *The optimistic FL protocol is per-packet secure.*

Proof. Eve can win the security game by causing a failure and either (a) convincing Alice that no failure occurred, or (b) causes Alice to localize a node that is not adjacent to Eve. We show that both (a) and (b) happen with negligible probability:

Consider (a) first. Recall that the packets d Alice sends in the security game are unique, and that each ack for a packet d contains the packet d. It follows that if Eve creates a valid ack to a packet d that was dropped before it arrived at Bob, she needs to forge the MAC on a message (B, d). From the security of the MAC, she can do this with only negligible probability.

Next, consider (b). Let R_i be the upstream-most node where Eve either caused a failure or tampered with an ack. We have two cases:

- Suppose all the nodes upstream of Eve's node R_i do not deviate from the correct algorithm. Let R_j be the first honest node that is downstream of node R_i (we know such a node exists because Eve cannot occupy Bob's node). Since R_{i-1} and R_j are honest, they correctly generate their onion reports θ_{i-1}, θ_j, and these reports must have different entries in their "data" fields (if Eve tampered with the packet at node R_i), and/or different entries in their "ack" fields (if Eve tampered with the ack at R_i). Now, since all the nodes upstream of R_{i-1} behave honestly, their onion reports are all be consistent. Also, conditioned on Eve not forging R_j's MAC on the onion report, we know that R_j's onion report is inconsistent. It follows that the upstream most transition from consistent to inconsistent reports must occur on some link between R_{i-1} and R_j and Alice will output a link adjacent to Eve.

- Suppose one of the nodes upstream of Eve's node R_i does deviate from the correct algorithm. Call the upstream-most such node R_e, and observe that it must be occupied by Eve. By the way we chose R_i, we know that R_e did not cause a failure or tamper with an ack. It follows that R_e must have tampered with an onion report request or an onion

[5] When each node originally receives the onion report request q from Alice, each node sets an local time-out that determines how long he should wait for his downstream neighbor to send their onion report. If the onion time-out expires, the node reports a missing onion report by setting $\theta_{i+1} = \perp$ and then proceeding to construct his own onion report as before.

report. Let R_j be the first honest node downstream of R_e. Conditioned on not forging the MAC of an honest node in the onion report, it follows that Eve at R_e must have caused a consistent/inconsistent transition at some link between R_{e-1} and R_j, and so that Alice will output a link adjacent to Eve.

Combining these two cases, we see that from the security of the MAC, (b) happens with negligible probability. □

Efficiency. We remark that the detect phase of this protocol incurs a high storage and communication overhead at each node on the path; we require the addition of at least one new $O(n)$-length message for each data packet sent, and even more egregiously, each node must store (an $O(n)$-length digest of) each packet it sends until it receives an ack or onion report request. This high overhead makes this protocol highly impractical for regular Internet traffic; however, it might be useful for specialized highly-secure networks, or for certain classes of low-volume traffic *e.g.,* network management traffic.

3.2 A Composition Technique for Statistical FL

We now consider statistical security protocols, that apply results from our previous work on statistical PQM [13, 12] to obtain statistical FL protocols with much lower overhead. In a statistical PQM protocol, Alice *detects* whenever the average failure rate exceeds a threshold β (but she need not localize a link).

Here we show how to compose the lightweight PQM protocols we presented in [13, 12] to obtain statistical FL protocols. While it is possible to give a very general composition theorem, for clarity and concreteness, we first describe how to compose the simpler *symmetric secure sampling (SSS)* protocol of [13,12] to obtain a protocol with storage and communication overhead that linear in (*i.e.,* a small fraction of) the number of sent packets in the interval, T. The protocol we present here requires each node to share *pairwise* keys with Alice and Bob. However, we can extend this result to the public-key setting by composing instances of the asymmetric secure sampling protocols of [13, 12] , to obtain a protocol that requires only a single public-key cryptographic operation per interval of T sent packets. For brevity, we omit any further discussion of this protocol here.

Finally, we show how to compose the *secure sketch protocol* of [13, 12] to obtain a more efficient FL protocol with about $O(K^2 \log T + n)$ storage overhead at each node and only two additional control messages.

3.2.1 A composition with that uses Secure Sampling PQM.

Symmetric Secure Sampling (SSS), a statistical PQM protocol from [13, 12] . SSS requires Alice and Bob to securely designate a random p fraction of the data packets that Alice sends to Bob as "probes", and require that Bob send MAC'd acknowledgments for all the probes. We call p the *probe frequency.* To do this, Alice and Bob share a secret $k = (k_1, k_2)$. For each packet d that Alice sends to Bob, they use k_1 to compute a function Probe that determines whether or not a packet d is a probe and should therefore be stored, and acknowledged. To acknowledge a probe, Bob sends Alice an ack $[d]_{k_2}$ that is MAC'd using k_2. The Probe function is implemented using a pseudorandom function (PRF) f keyed with k_1, that we think of as mapping strings to integers in $[0, 2^{n-1}]$; We define

$$\begin{array}{ll} \mathsf{Probe}_{k_1}(d) = \text{Yes} & \text{if } \frac{f_{k_1}(d)}{2^n} < p, \\ \mathsf{Probe}_{k_1}(d) = \text{No} & \text{otherwise.} \end{array} \tag{1}$$

For each interval, Alice stores each probe packet (*i.e.,* each packet d such that $\mathsf{Probe}_{k_1}(d)$ =Yes). At the end of the interval, after T packets are sent, Alice computes V, a count of the number

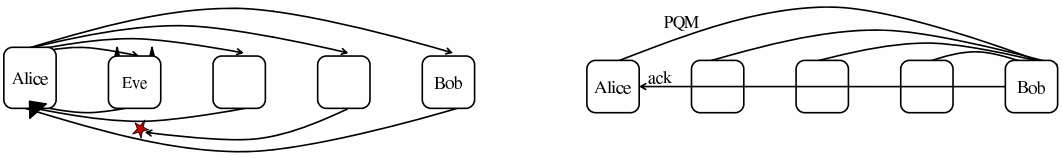

Figure 2: On the left an insecure composition, on the right our secure composition.

of stored (probe) packets for which she failed to receive a valid ack. She computes the average failure rate as $\frac{V}{pT}$.

A composition that does not work. Perhaps the most natural approach to construct a statistical FL protocol is to have Alice run K simultaneous PQM protocols with each of the intermediate nodes, and use the statistics from each protocol to infer behavior at each link (similar to [27,5,29]). However, we now show that this composition is vulnerable to the following *timing attack*: Suppose a packet d that Alice sends to Bob is ack'd by innocent node R_j with message a. Then, if Eve occupies node R_i for $i < j - 1$, she can determine that R_j originated the ack a by counting the time steps that elapsed between the time step in which she saw d and time step in which she saw a. Then, Eve can implicate R_j by selectively dropping every ack that originates at R_j. Notice that this attack results from the structure of this composition, and cannot be prevented even when acks are encrypted.[6] In practice, this attack can be launched when isolated burst of packets triggers a separate burst of acks at each intermediate node.

Composing PQM to statistical FL. We require that every node R_i shares pairwise keys k_i^A, k_i^B with Alice and Bob respectively. Using k_i^B, each intermediate node runs a statistical PQM protocol with Bob with the following modification: whenever Bob decides to send an ack for a packet d to an intermediate node R_i, Bob will (1) always address the ack to Alice and (2) MAC the ack in onion fashion, starting with k^B_{Alice} (on the inside of the onion) and ending with k_K^B (on the outside of the onion). Each node forwards all acks upstream, and processes only the ack he expects. At the end of the interval u, Alice will send an onion report request $q = (\mathsf{report}, u)$ to all the intermediate nodes. Each intermediate node produces a MAC'd onion report $\theta_i = [q, i, V_i, \theta_{i+1}]_{k_i^A}$ where V_i is his estimate of the average failure rate on the path between himself and Bob. Letting α, β be the false alarm and detection thresholds, when Alice receives the final onion report θ_1, she computes $F_\ell = V_i - V_{i+1}$ for each link $\ell = (i, i+1)$, and outputs ℓ if $F_\ell > \frac{\alpha+\beta}{2(K+1)}$, or if $\ell = (i, i+1)$ is the upstream-most link when the onion report θ_{i+1} refers to the wrong interval, is missing, or is invalidly MAC'd.
We prove that this scheme is secure provided that the interval length T is long enough and the congestion rate ρ is small enough.

Theorem 3.2. *The composition of SSS described above with probe frequency p satisfies (α, β, δ)-strong statistical security when each interval contains at least $T = O(\frac{K^2}{p(\beta-\alpha)^2} \ln \frac{K}{\delta})$ packets and the congestion rate satisfies $\beta - \alpha \gg K\rho$.*

Proof. First, observe that the probability that any efficient adversary Eve successfully forges an ack for a dropped packet by forging a MAC used in SSS is negligible. As in the Optimistic Protocol, the probability that any efficient adversary Eve successfully forges the onion report of an honest node (by forging the MAC on the onion report) is negligible as well. Hence, for the rest of this proof assume that Eve does not forge an ack to a dropped packet or validly forge the onion report of an honest node. Moreover, we can assume that Eve does not tamper with the onion report, or else she will implicate a link adjacent to one of the nodes she controls. We now work within a single interval:

- Let V_i be R_i's *estimate* of the failure rate between R_i and Bob.

[6] In [29], the authors suggest randomizing the sending time of acks.

- Let D_i be a count of the number of packets that were dropped or modified on the path between R_i and Bob.
- Let C_i be the number of acks intended for *any node* that were dropped or modified on the path between Bob and R_i.
- Let $p' = \frac{p}{1-(1-p)^{K+1}}$ be the probability that a node R_i expects an ack to a packet d (*i.e.,* $\mathsf{Probe}_{k_i^B}(d) = \text{Yes}$) conditioned on there being at least one node expecting an ack to packet d (*i.e.,* $\exists j \in \{0, \ldots, K\},\ \mathsf{Probe}_{k_j^B}(d) = \text{Yes}$).[7]

Note that when R_i estimates the average failure rate on the path from R_i to Bob, she is unable to distinguish between dropped packets and dropped acks. Also, it is possible that $D_i > D_{i+1}$ or $C_i > C_{i+1}$ for two adjacent uncorrupted nodes because of congestion. In the absence of adversarial behavior at R_i, the expectation of the estimator V_i that Alice receives in the onion report is $\frac{1}{T}(D_i + \frac{p'}{p}C_i)$. Finally, notice that the average failure rate on link $(i, i+1)$ is $\frac{1}{T}(D_i - D_{i+1})$.

Set $\gamma = \frac{\beta-\alpha}{2(K+1)}$. If $T = O(\frac{K^2}{p(\beta-\alpha)^2} \ln \frac{K}{\delta})$ then we have the following lemmata:

Lemma 3.3 (Deviation of the estimator V_i)**.** *For each $i \notin E$ where E is the set of nodes corrupted by Eve it holds (up to negligible error) that*

$$\Pr\left[\left|V_i - \tfrac{1}{T}(D_i + \tfrac{p'}{p}C_i)\right| > \tfrac{1}{4}\gamma\right] < \tfrac{\delta}{4(K+1)}$$

Lemma 3.4 (Acks dropped due to congestion)**.** *For each $i, i+1 \notin E$, it holds (up to negligible error) that*

$$\Pr\left[\tfrac{p'}{p}\tfrac{C_i - C_{i+1}}{T} > \tfrac{\gamma}{2}\right] < \tfrac{\delta}{2(K+1)}$$

The proofs of these lemmata are technical, but not difficult. We defer them to Appendix B.1. Both proofs are applications of the Chernoff bound under the assumption that the Probe function is implemented with a truly random function; the negligible error refers the difference between a PRF and a truly random function. The proof of Lemma 3.3 relies on the fact that Eve cannot bias node R_i's estimate of C_i by selectively dropping acks because (1) acks destined for different nodes look identical, and they all originate at Bob (so that an adversary cannot use timing to distinguish between them), and (2) acks are onion MAC'd, so the adversary cannot selectively tamper with an ack intended for an upstream node. The proof of Lemma 3.4 also relies on the fact that $\beta - \alpha \gg K\rho$.

Few false positives: To prove this, we consider an interval where all the nodes on the path behave honestly, and show that, with probability at least $1 - \delta$, Alice will not raise an alarm during this "honest interval".

Consider link $\ell = (i, i+1)$ where the average failure rate is less than the false alarm threshold so $\frac{1}{T}(D_i - D_{i+1}) < \frac{\alpha}{K+1}$. We now show that Alice will not raise an alarm for this link ℓ by proving that Alice's estimate of the failure rate for ℓ, *i.e.,* $V_i - V_{i+1}$, does not exceed her alarm decision threshold, *i.e.,* $\frac{\alpha+\beta}{2(K+1)}$. We do this by proving that

$$\Pr\left[\left|(V_i - V_{i+1}) - \tfrac{1}{T}(D_i - D_{i+1})\right| > \tfrac{\alpha+\beta}{2(K+1)} - \tfrac{\alpha}{K+1} = \gamma\right] < \tfrac{\delta}{K+1} \tag{2}$$

Notice that "Few false positives" condition follows from (2) by a union bound over all $K+1$ links.

[7] This quantity is the probability that a node R_i samples an ack that was dropped between R_i and R_B, since at least one node must have sampled the corresponding packet in order for the ack to be transmitted at all.

To prove (2), we start with the expression below, and apply the triangle inequality, and then Lemma 3.3:

$$\begin{aligned}
&\Pr[|(V_i - V_{i+1}) - (\tfrac{D_i - D_{i+1}}{T} + \tfrac{p'}{p}\tfrac{C_i - C_{i+1}}{T})| > \gamma/2] \\
&\quad \leq \Pr[|V_i - \tfrac{1}{T}(D_i + \tfrac{p'}{p}C_i)| > \gamma/4] + \Pr[|V_{i+1} - \tfrac{1}{T}(D_{i+1} + \tfrac{p'}{p}C_{i+1})| > \gamma/4] \\
&\quad \leq \tfrac{\delta}{2(K+1)} \qquad (3)
\end{aligned}$$

Next, from Lemma 3.4 we know that $\Pr[\frac{p'}{p}\frac{C_i - C_{i+1}}{T} > \gamma/2] \leq \frac{\delta}{2(K+1)}$, and so a union bound over this expression and (3) proves (2).

Secure localization: We now show that if Eve drops more than a β fraction of packets in any interval, then Alice will catch her with probability at least $1 - \delta$. Since the actual failure rate on the path is $\frac{1}{T}D_A > \beta$, we start by applying Lemma 3.3 to find that Alice's estimate of the failure rate is $V_A > \beta - \frac{\gamma}{4}$ with probability at least $1 - \frac{\delta}{4(K+1)}$. We now use an averaging argument to claim that there exists some link $\ell = (i, i+1)$ such that $V_i - V_{i+1} > \frac{\alpha+\beta}{2(K+1)}$. To see why, suppose for the sake of contradiction that for all i we had $V_i - V_{i+1} \leq \frac{\alpha+\beta}{2(K+1)}$. Then, it follows that

$$V_A = \sum_{i=0}^{K}(V_i - V_{i+1}) \leq \sum_{\ell} \tfrac{\alpha+\beta}{2(K+1)} = \tfrac{\alpha+\beta}{2} < \beta - \tfrac{\gamma}{4}$$

where $V_{K+1} = 0$ (Bob's estimate of drops to himself is 0). But this contradicts our condition that $V_A > \beta - \frac{\gamma}{4}$, so there is at least one link $\ell = (i, i+1)$ with $V_i - V_{i+1} > \frac{\alpha+\beta}{2(K+1)}$ so that Alice raises an alarm.

Next, recall that we assume that for any link where the true failure rate due to congestion less than $\frac{\alpha}{K+1}$, we have from our proof of the "Few false positives" condition that with probability $\frac{\delta}{K+1}$, Alice does not raise an alarm for link ℓ between two honest nodes. Then, Alice must have raised the alarm for a link adjacent to Eve with probability at least $1 - \delta$ (by a union bound) or a link with actual failure rate larger than $\frac{\alpha}{K+1}$, and secure localization follows. □

Efficiency. We remark that this protocol requires Alice and each intermediate node to store tags of length $O(n)$ for a p-fraction of the packets that they send. The communication overhead of the protocols is similarly a p-fraction of $O(n)$-length tags. Notice that, under the assumption that the interval T is long enough, we can take p to be arbitrarily small.

3.2.2 A composition that uses Secure Sketch PQM

Secure Sketch, a statistical PQM protocol from [13, 12] . In Secure Sketch PQM, Alice and Bob to securely aggregate information about all the traffic that Alice sends to Bob in short hash-based data-structure called a sketch. At the end of the interval, Alice and Bob exchange their sketches using in MAC'd control messages, and use the sketches to estimate the failure rate on the path. To do this, Alice and Bob share a secret $k = (k_1, k_2)$. The key k_1 is used to key a PRF that is used to at the beginning of interval u by both parties to derive the interval key k_u as $k_u = f_{k_1}(u)$. For each data packet d, Alice and Bob use the interval key k_u to compute a hash $f_{k_1}(d)$ of the packet. The output of the hash function is a length N-vector that added to a vector of N counters, each of length b, called the sketch. After T packets are sent and the interval ends, Alice sends Bob control message that contains her sketch and the next interval number, and is MAC'd with k_2. Bob responds by subtracting Alice's sketch from his own, and replying with a MAC'd control message containing the interval number and difference between the two sketches. Alice then obtains an estimate of the failure rate V by computing some function g on the difference between the two sketches.

The security of secure sketch PQM. Briefly, the secure sketch protocol works because it correctly estimates the p^{th}-moment of a packet stream (for some $p \geq 1$). That is, consider the stream of T packets that Alice sends to Bob during the interval, where each packet is chosen from a universe U (*e.g.,* if packets are 1500bytes, then $|U| \approx 2^{1500 \cdot 8}$). Let $\mathbf{v}_A$ be the characteristic vector of this stream, a U-dimensional vector that has c in the position corresponding to packet x if packet x was sent c times during the interval. Similarly, let $\mathbf{v}_B$ be the characteristic vector of the stream of packets received by Bob. Then, the sketches allow Alice to estimate $\|\mathbf{v}_A - \mathbf{v}_B\|_p$. In particular, we say that a sketching protocol (ε, δ)-estimates the p^{th}-moment of the characteristic vector $\mathbf{v}_A - \mathbf{v}_B$ if

$$\Pr\left[\left|V - \|\mathbf{v}_A - \mathbf{v}_B\|_p\right| \leq \varepsilon \|\mathbf{v}_A - \mathbf{v}_B\|_p\right] < 1 - \delta \tag{4}$$

where the probability is taken over the randomly chosen key k_u used to key the packet-hash function f. In [13, 12] , we discuss exactly how to choose the hash function f, and how this choice affects p, the norm estimated by the sketch, and $N \times b$, the size of the sketch. For our purposes we shall simply note that if the hash function f is an appropriately-chosen PRF, then we can use sketch of size $N \times b$ where $N = O(\frac{1}{\varepsilon^2}\log(\frac{1}{\delta}))$ and $b = O(\log(T))$.

We no longer have timing attacks. In secure sketch PQM, Alice and Bob exchange only a pair of control messages at the end of the interval; no other communication between them is required. Because the *timing* of these control messages do *not* leak any information, the timing attack we mentioned in Section 3.2.1 is no longer an issue. Our composition of secure sketch PQM to statistical FL will have Alice run K simultaneous PQM protocols with each of the intermediate nodes as in Figure 2, and use the statistics from each protocol to infer behavior at each link.

A simpler composition. We require that every node R_i shares pairwise keys k_i with Alice only (*c.f.,* with our SSS-based composition, where nodes need to share keys with Bob as well). Using k_i, each intermediate node runs a secure sketch PQM protocol with Alice, so that Alice will keep a sketch $\mathbf{w}_i^A$ for every $i \in [K]$ and every other node R_i will keep a single sketch $\mathbf{w}_i$. However, instead of sending individual control messages to each node at the end of interval u, Alice will now send a single onion-MAC'd interval-end message containing all her sketches as

$$q = [(u, \mathbf{w}_1^A)[(u, \mathbf{w}_2^A)...[(u, \mathbf{w}_B^A)]_{k_B}...]_{k_2}]_{k_1}$$

to all the intermediate nodes. Upon receiving a validly-MAC'd interval-end message, intermediate node R_i extracts the sketch w_i^A, and passes the interval-end message to R_{i+1}. (R_i drops the interval-end message if the MAC is invalid.) Finally, as in the usual composition, each node R_i produces a MAC'd onion report $\theta_i = [u, i, V_i, \theta_{i+1}]_{k_i}$. Here, V_i is node R_i's estimate of $\|\mathbf{v}_i - \mathbf{v}_A\|_p$, which is computed by applying the function g to the difference sketch $\mathbf{w}_i - \mathbf{w}_i^A$. (Recall that v_A is the characteristic vector of the stream of packets that Alice sends, and $\mathbf{v}_i$ is the characteristic vector of the stream of packets that R_i receives.) Letting α, β be the false alarm and detection thresholds, when Alice receives the final onion report θ_1, computes $F_\ell = V_i - V_{i+1}$ for each link $\ell = (i, i+1)$, and outputs ℓ if $\ell = (i, i+1)$ is the upstream-most link where $F_\ell > \frac{T}{K+1}\frac{\beta(2\alpha+\beta)}{\alpha+2\beta} = \Gamma$, or the onion report θ_{i+1} refers to the wrong interval, is missing, or is invalidly MAC'd. If there is no such link, she outputs $\surd$.

Limiting the number of nodes occupied by Eve. To prove that this scheme is secure, we need to assume that interval length T is long enough, the sketches are big enough, and the congestion rate ρ is small enough. Our proof also relies on limiting the number of links occupied by Eve to $\approx \sqrt{K}$. However, we conjecture that it may be possible to weaken this assumption, as we have not been able to find any attacks on the protocol when Eve occupies more than $\sqrt{K}$ links. For more discussion, see the remarks in Appendix B.4.

Theorem 3.5. *The composition of secure sketch PQM described above satisfies (α, β, δ)-statistical security if the congestion rate satisfies $\rho K^2 \leq \beta$, Eve occupies $M \leq \sqrt{(K+1)(1 - \frac{\rho}{\beta}K^2)}$ links,*

each interval contains at least $T > \frac{K+1}{\alpha}$ *packets, and for each* $i \in [K]$, *sketches* $\mathbf{w}_i, \mathbf{w}_i^A$ *have size*

$$N_i \times b = O\left(i^2 \left(\frac{2\beta+\alpha}{\beta-\alpha}\right)^2 \log(\tfrac{K}{\delta})\right) \times O(\log T) \tag{5}$$

Proof. First, the probability that any efficient adversary Eve successfully forges the interval end message or the onion report of an honest node (by forging the MAC) is negligible. Hence, for the rest of this proof assume that Eve does not validly forge the onion report of an honest node. Moreover, we can assume that Eve does not tamper with the interval-end message of the onion report, or else she will implicate a link adjacent to one of the nodes she controls. We now work within a single interval, and use the following definitions:

- Let $\mathbf{v}_A$ is the characteristic vector of the stream of packets that Alice sends and $\mathbf{v}_i$ for $i \in [K+1]$ to be the characteristic vector of the stream of data packets that R_i receives.
- Let $\mathbf{x}_i = \mathbf{v}_i - \mathbf{v}_A$. We can decompose any $\mathbf{x}_i$ into two vectors $\mathbf{x}_i = \mathbf{d}_i + \mathbf{a}_i$. The vector $\mathbf{d}_i$ is the characteristic vector of packets *dropped* on the path from Alice to R_i, and contains the non-negative components of $\mathbf{x}_i$. The vector $\mathbf{a}$ is the characteristic vector of packets *added* on the path from Alice to R_i, and contains the non-positive components of $\mathbf{x}_i$. Also notice that the non-zero coordinates of $\mathbf{d}$ and $\mathbf{a}$ are disjoint.
- Let V_i be R_i's *estimate* of $||\mathbf{x}_i||_p^p$.
- Let D_i be a count of the number of failures that occurred on the path between Alice and R_i.

Our proof also makes use of the following identity

$$||\mathbf{x}_i||_p^p = ||\mathbf{d}_i||_p^p + ||\mathbf{a}_i||_p^p = D_i + ||\mathbf{a}_i||_p^p \tag{6}$$

The first equality follows because the non-zero coordinates of $\mathbf{d}$ and $\mathbf{a}$ are disjoint. The second equality follows because every packet that Alice send is unique so that that $\mathbf{d}$ is a $\{0,1\}$-vector for every $i \in [K+1]$. In [13, 12] we show that if interval key is refreshed at the end of each interval, then if each sketch has $N_i \times b = O(\frac{1}{\varepsilon_i} \log \frac{1}{\delta'})$ then it follows that each estimate V_i (ε_i, δ')-approximates $||\mathbf{x}_i||_p^p$ as per (4). Also, we will require that $\frac{\alpha}{K+1} T > 1$ (which gives us the bound on T, the number of packets in the interval), and prove the following lemma in Appendix B.3:

Lemma 3.6. *Let* $\Gamma = \frac{T}{K+1} \frac{\beta(2\alpha+\beta)}{\alpha+2\beta}$ *and* $\varepsilon_i = \frac{1}{2i} \frac{\beta-\alpha}{2\beta+\alpha}$. *For every* $i \in [K]$, *assume that* R_i *computes an estimate* V_i *that* (ε_i, δ')*-estimates* $||\mathbf{x}_i||_p^p$. *Suppose also that* $||\mathbf{x}_i||_p^p \leq \frac{\beta i}{K+1}$. *Then with probability at least* $1 - 2\delta'$ *it follows that:*

1. *If "link* $(i, i+1)$ *is good" so that* $||\mathbf{x}_{i+1}||_p^p - ||\mathbf{x}_i||_p^p \leq \frac{\alpha}{K+1} T$ *then* $V_{i+1} - V_i \leq \Gamma$.
2. *If "link* $(i, i+1)$ *is bad" so that* $||\mathbf{x}_{i+1}||_p^p - ||\mathbf{x}_i||_p^p \geq \frac{\beta}{K+1} T$ *then* $V_{i+1} - V_i \geq \Gamma$.

We use Lemma 3.6 to prove the "few false positives" and "secure localization" conditions.

Few false positives: To prove this, we consider an interval where all the nodes on the path behave honestly. During this interval, we know that no packets were added anywhere on the path (so that $||\mathbf{a}_i||_p^p = 0$ for each $i \in [K+1]$) and less than $\frac{\alpha}{K+1}$ packets were dropped at each link. We can apply identity (6) to find that for each link $(i, i+1)$ we have

$$||\mathbf{x}_{i+1}||_p^p - ||\mathbf{x}_i||_p^p = D_{i+1} - D_i + 0 + 0 \leq \tfrac{\alpha}{K+1} \tag{7}$$

and the telescoping nature of (7) gives us that

$$||\mathbf{x}_i||_p^p = (||\mathbf{x}_i||_p^p - ||\mathbf{x}_{i-1}||_p^p) + \ldots + (||\mathbf{x}_2||_p^p - ||\mathbf{x}_1||_p^p) + ||\mathbf{x}_1||_p^p \leq \tfrac{\alpha i}{K+1} \tag{8}$$

We can now apply Lemma 3.6 to show that, with probability at least $1-2\delta'$ we have that $V_{i+1} - V_i \leq \Gamma$ so that Alice will not output link $(i, i+1)$. A union bound over the $K+1$ links gives us that Alice will output $\surd$ during this interval with probability at least $1-2(K+1)\delta_i$.

Secure localization: We now show that if Eve causes more than a β fraction of failures in the interval, then with probability at least $1-\delta$, Alice will either catch Eve or output a link with more than $\frac{\alpha}{K+1}$ failures. Recall that Alice outputs the upstream-most link $\ell = (i, i+1)$ for which there is an "alarm", *i.e.,* where $V_{i+1} - V_i \geq \Gamma$. We need the following simple observation:

Lemma 3.7. *Define event E_i as the event that $\|\mathbf{x}_i\|_p^p \leq \frac{\beta i}{K+1}$ .For each $i \in [K+1]$, if Alice does not raise an alarm for any link upstream of link i, then E_i holds with probability $1-2i\delta'$.*

Proof. Suppose that Alice does *not* raise an alarm for all the links upstream of node R_i. It follows from Lemma 3.6 that $\|\mathbf{x}_{j+1}\|_p^p - \|\mathbf{x}_j\|_p^p \leq \frac{\beta}{K+1}$ with probability $1-2\delta'$, for each link $(j, j+1)$ where $j \in [i-1]$. The lemma follows by taking a union bound over all these links and using a telescoping sum as in (8). □

First we show that the with high probability Alice will not output an honest link. Let link $(i, i+1)$ is be "honest", *i.e.,* have a fewer than $\frac{\alpha}{K+1}$ failures, and assume that Alice does not raise alarm for any links upstream of R_i. Now, Lemma 3.6 shows that, conditioned on E_i, Alice will not raise an alarm for link $(i, i+1)$ with probability at least $1-2\delta'$. Since Alice does not alarm for any links upstream of R_i, we can apply Lemma 3.7 to remove the conditioning on E_i. It follows that Alice will not output honest link $(i, i+1)$ with probability at least $1-2(i+1)\delta'$. Taking a union bound over all honest links gives that Alice will not alarm for any honest link with probability at least $1-2(K+1)^2\delta'$.

Next, we need to show that Alice either will raise an alarm for a link adjacent to Eve or link with more than $\frac{\alpha}{K+1}$ failures. The most interesting part of this proof is the following technical lemma, which we prove in Appendix B.4:

Lemma 3.8. *If Eve occupies $M \leq \sqrt{(K+1)(1-\frac{\rho}{\beta}K^2)}$ links and causes a β-fraction of failures in the interval, then there must be a link $(i, i+1)$ that is adjacent to Eve with*

$$\|\mathbf{x}_{i+1}\|_p^p - \|\mathbf{x}_i\|_p^p \geq \frac{\beta}{K+1}T$$

Now let link $(i, i+1)$ be the upstream-most link that is adjacent to Eve and has $\|\mathbf{x}_{i+1}\|_p^p - \|\mathbf{x}_i\|_p^p \geq \frac{\beta}{K+1}T$. (Lemma 3.8 guarantees the existence of such a link.) We have two cases:

- Suppose Alice did not raise an alarm for a link upstream of R_i. Combining Lemma 3.7 and Lemma 3.6 it follows that Alice will alarm for link $(i, i+1)$ adjacent to Eve with probability $1-2(i+1)\delta'$.

- Suppose Alice did raise an alarm for a link upstream of R_i. It follows from Lemma 3.6 that there is some link $(j, j+1)$ for $j \leq [i-1]$ where, with probability $1-2\delta'$,

  $$\frac{\alpha}{K+1} \leq \|\mathbf{x}_{j+1}\|_p^p - \|\mathbf{x}_j\|_p^p = D_{j+1} - D_j + \|\mathbf{a}_{j+1}\|_p^p - \|\mathbf{a}_j\|_p^p$$

  where the equality comes from applying identity (6). Now if link $(j, j+1)$ is adjacent to Eve, it follows that Alice alarms for a link adjacent to Eve, and we are done. Thus, suppose that link $(j, j+1)$ is *not* adjacent to Eve. Then, it follows that no new packets could have been added to this link, and so we have that $\|\mathbf{a}_{j+1}\|_p^p = \|\mathbf{a}_j\|_p^p$. Thus, if link $(j, j+1)$ is *not* adjacent to Eve, then Alice must have raised an alarm for a link with $D_{j+1} - D_j \geq \frac{\alpha}{K+1}$ failures, as required.

Combining these cases, we see that with probability at least $1 - 2(K+1)\delta'$, Alice will either raise an alarm for a link that is either (a) adjacent to Eve, or (b) has more than $\frac{\alpha}{K+1}$ failures.

Sizing the sketches. Finally, to ensure that (α, β, δ)-statistical security holds, it suffices to take $\delta' = \delta/4(K+1)^2$. Next, recall that Lemma 3.6 requires sketches that (ε_i, δ')-estimate the p^{th} moment with $\varepsilon_i = \frac{1}{2i}\frac{\beta-\alpha}{2\beta+\alpha}$. For $i \in [K+1]$ it suffices to take sketches $\mathbf{w}_i$, $\mathbf{w}_i^A$ of size $N_i \times b$ where $N_i = O(\frac{1}{\varepsilon_i^2}\log(\frac{1}{\delta'}))$ and $b = O(\log(T))$. Substituting in the values for ε_i, δ' gives us (5) as required. □

Efficiency. We remark that, for a given interval of length T, this protocol requires $O(K^2 \log T + n)$ storage overhead at Bob and each intermediate node, while the storage overhead at Alice is $O(K^3 \log T + Kn)$. The communication overhead of the protocol is two control messages of length $O(K^3 \log T + Kn)$ each for every T packets sent.

4 Lower bounds

We now argue that in any secure per-packet FL scheme Alice requires shared keys with Bob and the intermediate nodes, and Alice, Bob and each intermediate node must perform cryptographic operations. We only argue for intermediate nodes $R_2, \ldots, R_K$; R_1 is a border case which requires neither keys nor crypto because we assume Alice is always honest.

4.1 Failure Localization Needs Keys at Each Node

Since FL provides strictly stronger security guarantees than path-quality monitoring, it follows from the results in [13, 12] that in any secure FL protocol, Alice and Bob must have shared keys. We also have the following theorem that proves that in any secure FL protocol, *each* intermediate node must share keys with some Alice:

Theorem 4.1. *Suppose* Init *generates some auxiliary information* aux_i *for each node* R_i *for* $i = 1, ..., K, \mathsf{Alice}, \mathsf{Bob}$. *A FL protocol cannot be (per-packet or statistical) secure if there is any node* $i \in \{2, \ldots, K\}$ *such that* $(\mathsf{aux}_{\mathsf{Alice}}, \mathsf{aux}_1, \ldots, \mathsf{aux}_{i-1})$ *and* aux_i *are independent.*

Proof. Suppose R_i has aux_i that is independent of $(\mathsf{aux}_{\mathsf{Alice}}, \ldots, \mathsf{aux}_{i-1})$. Then, the following two cases are indistinguishable from Alice's view: (a) Node $R_{i+1}$is malicious and blocks communication on link $(i, i+1)$, and (b) Eve occupies node R_{i-1}, and drops packets while simulating case (a) by picking an independent aux_i' and running $R_i(\mathsf{aux}_i')$ while pretending as if $(i, i+1)$ is down. These two cases are indistinguishable because aux_i is independent of $(\mathsf{aux}_{\mathsf{Alice}}, \ldots, \mathsf{aux}_{i-1})$, and so Alice will localize the failure to the same link in both case (a) and (b). But this breaks security, since R_{i+1}, R_{i-1} do not share a common link. □

4.2 Failure Localization Needs Crypto at Each Node

In [13,12] , we give a reduction from one-way functions to secure PQM, proving:

Theorem 4.2 (From [13,12])**.** *The existence of a per-packet secure PQM protocol implies the existence of an infinitely-often one-way function (i.o.-OWF).*

Since one-way functions are equivalent to many cryptographic primitives (in the sense that these primitives exist if and only if one-way functions exist [18]), this result can be interpreted to mean that nodes participating in any secure PQM protocol must perform cryptographic computations. Since FL gives a strictly stronger security guarantee than PQM, we also have that in any FL protocol, some node on the data path must perform cryptography. However, Theorem 4.2 only implies that the *entire system* performs cryptography. We want to prove that any secure FL protocol requires *each intermediate node* $R_1, \ldots, R_K$ to perform cryptography. Because it

is not clear even how to formalize this in full generality, we instead apply the methodology of Impagliazzo and Rudich [19] to do this for *black-box* constructions of FL protocols from a random oracle RO. We model "performing cryptography" as querying the random oracle, and show that in such a secure FL protocol *each node* must query the RO.

In [19], Impagliazzo and Rudich showed that there can be no secure black-box construction of key agreement (KA) from a random oracle. They argued that if any such KA construction is secure, then it must also be secure in a relativized world where every party has access to a random oracle RO, and a PSPACE oracle. (A PSPACE oracle solves any 'PSPACE-complete problem, *e.g.,* True Quantified Boolean Formulae (TQBF).) Intuitively, in this (PSPACE, RO) world, every computation is easy to invert *except* for those computed by the RO. They obtain their result by showing, for every possible black-box construction of KA from a random oracle, that there exists an efficient algorithm (relative to (PSPACE, RO)) that breaks the security of KA. Using the same reasoning, any secure black-box FL protocol constructed from a RO must remain secure even relative to a (RO, PSPACE) oracle. Then, to obtain our result, it suffices to exhibit an efficient algorithm (relative to (PSPACE, RO)) that breaks security of any black-box FL protocol where one node does not call RO. We do this below.

We will use the notion of an *exchange* to denote a data packet and all the FL-protocol-related messages associated with that packet. Because our game is sequential (see Section 2), Alice's must decide to localize a link ℓ or output $\surd$ before the next exchange begins. Let $\langle R_{i-1}, R_i\rangle_j$ denote the distribution of all messages sent and received along link $(i-1, i)$ during the j'th exchange. We sometimes refer to these messages as a transcript for the j'th exchange. Because we allow the nodes to keep state, this distribution may depend on what happened in all previous exchanges, $\langle R_{i-1}, R_i\rangle_1, \ldots, \langle R_{i-1}, R_i\rangle_{j-1}$. We now prove that a per-packet FL protocol with $2r = O(\log n)$ messages per exchange must invoke the random oracle at every node. We assume that the number of messages per exchange is even, and that odd messages go from R_{i-1} to R_i and even messages go from R_i to R_{i-1}.We note that protocols where number of messages per packet grows with n are impractical and so "practical" protocols should use $2r = O(1)$ messages per exchange. (See Remark 4.7 below on the possibility of extending this result to statistical security and/or protocols with $\omega(\log n)$ messages per exchange.)

Theorem 4.3. *Fix a fully black-box per-packet FL protocol that uses access to a random oracle* RO*, where at least one node R_i for $i \in \{2, \ldots, I\}$ never calls the* RO *and where the maximum number of messages per exchange is $O(\log n)$. Then there exists an efficient algorithm relative to* (PSPACE, RO) *that breaks the security of the scheme with non-negligible probability over the randomness of* RO *and the internal randomness of the algorithm.*

The proof of Theorem 4.3 is quite technical and is deferred to Appendix C. We sketch the proof, which resembles that of Theorem 4.1. Eve controls node R_{i-1} and impersonates R_i, but now aux_i is secret, so Eve must first *learn* aux_i:

1. *Learning to impersonate.* Sitting at R_{i-1}, Eve observes t exchanges (t is polynomial in n), where Eve asks Source to transmit a uniformly random data packet. She then uses the learning algorithm of Naor and Rothblum [26] to obtain a pair of impersonator algorithms A', B', whose interaction generates a distribution over transcripts for the $t+1$'th exchange. A' impersonates nodes Alice, $R_1, \ldots, R_{i-1}$ and B' impersonates nodes $R_i, \ldots, R_K$, Bob.

2. *Dropping and impersonating.* On the $t+1$'th exchange, for each message m_j going from R_{i-1} to R_i, Eve computes a response herself m_{j+1} using algorithm B' and returns m_{j+1} to R_{i-1}; she does not send any messages to R_i. (More precisely, B' samples m_{j+1} according to the conditional distribution $\langle A', B'\rangle^{j+1} \mid \langle A', B'\rangle^j = (m_1, ..., m_j)$. Here $\langle A', B'\rangle^j$ denotes the first j messages of $\langle A', B'\rangle$. Note that this sampling is efficient in the presence of a PSPACE oracle.)

Now, Eve at R_{i-1} will break security if she manages to use B' to impersonate an *honest* exchange during which link $(i, i+1)$ is down. (This breaks security since link $(i, i+1)$ is not adjacent to R_{i-1}.) The crucial observation is that here, Eve need only impersonate node R_i, and that R_i does not "protect" its secret keys by calling the RO. Intuitively, Eve should be able to impersonate R_i since any computations that R_i does are easy to invert in the (PSPACE, RO) world. To prove the theorem, we shall show that with non-negligible probability $> (10/\rho)^r = 1/\text{poly}(n)$, the following are 1/100-indistinguishable: (a) Alice's view when link $(i, i+1)$ is down and (b) Alice's view when R_{i-1} drops a packet but impersonates link $(i, i+1)$ being down using B'.

In the following, we define the statistical distance between two random variables X, Y as $\Delta(X,Y) = \frac{1}{2}\sum_{x \in U} |\Pr[X = x] - \Pr[Y = x]|$ where U is the union of the supports of X and Y (for more background on statistical distance, see *e.g.,* [14]).

Recall (Section 2) that Alice is allowed to use information from past exchanges to help her decide how to send messages in new exchanges. Fortunately, the algorithm of Naor and Rothblum [26] is specifically designed to deal with this, and guarantees the following:

Lemma 4.4 (Based on [26]). *Relative to a* (PSPACE, RO)*-oracle, there exists an efficient algorithm that observes at most* $t = O(\frac{n}{\varepsilon^4})$ *honest exchanges* $\langle R_{i-1}, R_i \rangle_{1,...,t}$ *and then, with probability* $> 1 - \varepsilon$*, outputs efficient impersonator algorithms* $R'_0, \ldots, R'_{K+1}$ *such that that an impersonated transcript* $\langle R'_{i-1}, R'_i \rangle_{t+1}$ *(given by simulating the interaction of all the impersonator algorithms) for the exchange* $t+1$ *is distributed* ε*-close in statistical distance to the honest transcript* $\langle R_{i-1}, R_i \rangle_{t+1}$ *for exchange* $t+1$*.*

Suppose Eve obtained an A', B' where we let A' be the collection of algorithms $R'_0, \ldots, R'_{i-1}$ and B' be the collection $R'_i, \ldots, R'_{K+1}$that satisfy the guarantee above. Our first challenge is that the Naor-Rothblum algorithm does *not* guarantee that A', B' generates an impersonated transcript that is statistically close to the "honest" transcript of messages on $(i-1, i)$ when *the observer has access to the* RO. (The "honest" transcript of messages on the link $(i-1, i)$ is generated by interactions of honest Alice, $R_1, ..., R_K$, Bob.) Fortunately, with probability ρ^r all the messages sent from R_i to R_{i-1} are computed without access the RO. This happens when *congestion* causes link $(i, i+1)$ to go down for the duration of an exchange (so that R_i, who never calls the RO, has to compute all his upstream messages on his own).

Our next challenge is that Eve has no control, or even knowledge, of when congestion causes this event to occur. Indeed, the distribution generated by A', B' is only guaranteed to be close to the honest transcript overall; there is no guarantee that it is close to the honest transcript *conditioned on congestion on* $(i, i+1)$.[8] Fortunately, we can show that with probability ρ^r, A', B' will generate a "useful" impersonated transcript that is ε/ρ^r-statistically close to the honest transcripts conditioned on the event that link $(i, i+1)$ is down. Eve does not necessarily know *when* she impersonates a useful transcript; she simply has to hope that she is lucky enough for this to happen.

The last challenge is that even when Eve is lucky enough to obtain a useful transcript, we still need a guarantee that (a) conditioned on B' generating a useful transcript, using B' to *interact* with the honest algorithm R_{i-1} results in a transcript that is statistically close to (b) the transcript between honest algorithms R_{i-1} and R_i conditioned on link $(i, i+1)$ being down. Unfortunately, the Naor-Rothblum algorithm does not give any guarantees when an honest algorithm *interacts* with an impersonated algorithm for more than 1 round. Thus, we prove that, with probability at least $(\rho/2)^r$, the impersonator algorithm B' interacting with honest Alice, $...R_{i-1}$ still generates a useful transcript such that the statistical distance between (a) and (b) is at most 1/100. (This assumes we take ε small enough; $\varepsilon = (\rho/10)^{4r} = 1/\text{poly}(n)$ suffices.)

We address these challenges in the next lemma, which we prove in Appendix C. We state a general version of the lemma here, for which we first need a few definitions:

[8] For this reason, Eve cannot simply use R'_i (instead of $R'_i, ..., R'_K$, Bob′) to impersonate the honest R_i conditioned on link $(i, i+1)$ being down.

- Let A, B and A', B' be two (different) pairs of algorithms such that the statistical difference between the transcripts $\langle A, B\rangle$ and $\langle A', B'\rangle$ is bounded by ε. We assume *a priori* that A, B can share randomness, say by accessing a common random oracle, and so can A', B'.

- Let $(\langle A, B\rangle, \mathsf{view}_A(\langle A, B\rangle))$ be the joint distribution of transcripts $\langle A, B\rangle$ and the internal randomness of party A, which we call view_A (which includes both randomness that is shared with B and independent randomness). For a fixed τ, we will let $\mathsf{view}_A(\tau)$ be the distribution of the internal randomness of A conditioned on outputting the transcript τ.

- To deal with interaction, we let $\langle A, B'\rangle = (m_1, \ldots, m_r)$ be the distribution over transcripts where for each message m_j sent by A is computed honestly, while each m_j sent by B' is computed by *pretending that the partial transcript so far $\sigma_i = (m_1, \ldots, m_{j-1})$ came from the distribution $\langle A', B'\rangle$ and sampling the next message m_j consistent with $\langle A', B'\rangle$*; more formally B' samples m_j according to the conditional distribution $(\langle A', B'\rangle^j \mid \langle A', B'\rangle^{j-1} = \sigma_{j-1})$.[9] Here $\langle A', B'\rangle^j$ denotes the first j messages of $\langle A', B'\rangle$.

We are finally ready for the statement of the Lemma.

Lemma 4.5. *Suppose that $\Delta(\langle A, B\rangle, \langle A', B'\rangle) \le \varepsilon$, and there exist events $E_1, \ldots, E_r$ over the internal randomness of A, B such that (1) $\forall j$, conditioned on E_j, the first j messages from B to A are independent of A's internal randomness, and (2) $\Pr[E_j \mid E_{j-1}] \ge \rho$. Set $\varepsilon = (\rho/10)^{4r}$, Then there exist $\eta \ge (\rho/2)^r$, and distributions over the transcripts Y, Z such that $\langle A, B'\rangle$ is a convex combination $\eta Y + (1-\eta)Z$ and*

$$\Delta((Y, \mathsf{view}_A(Y)), (\langle A, B\rangle, \mathsf{view}_A(\langle A, B\rangle) \mid E_r)) \le 1/100$$

Lemma 4.5 tells us that, with probability η, $\langle A, B'\rangle$ will generate a "useful" transcript Y that is $\sqrt{\varepsilon}(10/\rho)^r$-statistically close to the honest transcript $\langle A, B\rangle$ conditioned on event E_r occurring. (Z is the "not useful" transcript that is generated with probability $1-\eta$.) We can now apply Lemma 4.5 by setting:

- A to be honest algorithms $R_0, R_1, \ldots, R_{i-1}$.

- B to be honest algorithms $R_i, \ldots, R_{K+1}$.

- A' to be the impersonator algorithms $R'_0, \ldots, R'_{i-1}$ given by Lemma 4.5.

- B' to be the impersonator algorithms $R'_i \ldots, R'_{K+1}$ given by Lemma 4.5.

- E_j to be the event that link $(i, i+1)$ is congested for the messages $1, \ldots, j$ that are sent downstream from B to A. (Then, E_r is the event that link $(i, i+1)$ is down for the duration of an exchange of length $r = O(\log n)$ messages.)

Now, notice that since R_i does not query the random oracle, conditioned on E_j the first j messages of B are independent of A because they are computed by R_i only. Next, note that $\Pr[E_j \mid E_{j-1}] = \rho$ because each message is lost to congestion independently.

To combine everything, set $\varepsilon = (\rho/10)^{4r}$ and apply Lemma 4.4 to find that with probability at least $\ge (1-\varepsilon)$ we get A', B' that is ε-close to $\langle A, B\rangle$ (notice that Eve is efficient with this setting of ε). Conditioned on this happening, by Lemma 4.5 we get that with probability $(\rho/2)^r = 1/\mathrm{poly}(n)$, Eve is lucky enough to generate a useful transcript such that (a) the view of Alice when Eve drops a packet at R_{i-1} and impersonates using $R'_i, \ldots, R'_{K+1}$ is 1/100-indistinguishable from the situation (b) where link $(i, i+1)$ is completely down for the duration of an exchange. Since Alice should localize the same link in case (a) and (b) for all but a 1/100 fraction of the time, this breaks security since link $(i, i+1)$ is not adjacent to Eve at R_{i-1}.

[9] In general this is not efficient, but it is efficient because in our setting Eve has access to a PSPACE oracle.

4.2.1 Statistical security.

Our lower bounds in the statistical setting are more subtle. First of all, from [13, 12] the analog of Theorem 4.2 also holds, showing that the *entire system* needs to "perform cryptography".

Theorem 4.6 (From [13, 12]). *The existence of a (α, β, δ)-statistically secure failure detection scheme for constants α, β, δ implies the existence of an infinitely-often one-way function (i.o.-OWF).*

However, we run into trouble when we try to show that cryptography is required at *each intermediate node*. It turns out that Definition 2.3 does *not* inherently require complexity-based cryptography at intermediate nodes. We sketch a statistically secure FL protocol where the intermediate nodes $R_1, \ldots, R_K$ use only information-theoretically secure primitives (although Alice and Bob still use regular MAC's). While this protocol is completely impractical in terms of communication and storage overhead, we present it here to demonstrate the subtleties of Definition 2.3.[10]

Remark 4.7 (Impractical "crypto-free" statistical FL protocol.)**.** *The protocol uses one-time MACs (OTMAC), information-theoretic objects that have the same properties as regular MACs except that they can only be used a single time. (OTMACs and can be constructed from universal hash functions [7].) Each node R_i shares pairwise keys with Alice. All the intermediate nodes and Bob store each packet that Alice sends to Bob. For each packet, Bob replies with an ack signed using a regular MAC. At the end of the interval, Alice counts the number of acks that she either fails to receive, or are invalid. The first time this count exceeds a β-fraction, Alice sends a "report request" message that is signed using a OTMAC to $R_1, \ldots, R_K, R_{K+1}$. Each node $R_1, \ldots, R_K$ responds with a report of every single packet they have witnessed, that is "onion signed" using the OTMAC (as in Section 3.1). Alice uses these reports in the usual way to localize link ℓ adjacent to Eve. From this point onwards Alice simply counts valid acknowledgments from Bob, and blames link ℓ each time the count exceeds a β fraction.*

The protocol satisfies Definition 2.3 because the probability that the failure rate at any link exceeds β by congestion alone is negligible. Since we do not allow Eve to move during the security game, if Alice successfully localizes Eve to link ℓ once, it means it must have been Eve's fault, and so from then on Alice can always blame all failures on link ℓ. As noted above, similar "impractical" protocols exist for per-packet protocols with $\omega(\log n)$ additional messages per packet (since all $\omega(\log n)$ messages are lost to congestion with only negligible probability), except that we replace the idea of "exceeding β fraction of failures" with "losing an entire exchange due to congestion". We may interpret this as follows:

1. It is unreasonable to assume that the failure rate at a link exceeds β only due to adversarial behavior (*i.e.,* Eve). For example, occasionally congestion might spike, or a router might malfunction or go down due maintenance, causing more than a β-fraction of packets to be dropped. If we assume such events happen with non-negligible probability, we can adapt the proof of Theorem 4.2 to show that cryptography is necessary at intermediate nodes for statistical security. As a corollary, if Eve can control congestion at links she does not occupy, then we need cryptography at every intermediate node. Our FL protocols remain secure even under the strongest such definition, where the failure rate on a link not occupied by Eve can exceed β.

[10] In concurrent work, Wong et al. [29] propose a statistical FL scheme where no cryptography is performed *during an interval*. Instead, they precompute shared secrets that are appended to packets over the course of an interval and are used guarantee security. The secrets must refreshed periodically, which requires cryptographic participation by the intermediate nodes. This contrasts with the impractical scheme we describe here, which truly *never* requires any intermediate node to perform crypto.

2. We can take this issue outside of our model. If we say that it is reasonable that Eve cannot move during the security game, and that the failure rate cannot exceed β on a link that Eve does not control, then, as we showed above, there exist protocols where the intermediate nodes do not use complexity-based cryptography. However, we must be cognizant that in the real world there can be multiple adversaries that we would like to localize correctly, or the adversary may be able to move from one link to another. If protocols that do not use cryptography at intermediate nodes are to remain secure after Eve moves (and learns the key of previous nodes she occupied), then the keys at each node should be refreshed periodically. This key refresh process would require each intermediate node to use cryptography.

5 Open problems

We gave lower bounds on the key-management and cryptographic overhead of secure FL protocols. While our statistical FL protocol based on sketching requires fairly small storage overhead, the interesting problem of bounding the *storage* requirements in an FL protocol is still open. Furthermore, our results here only apply to FL on *single symmetric paths* between a single sender-receiver pair. An interesting question would be to consider FL for *asymmetric paths*, where the packets Bob sends back to Alice may take a different path than the packets that Alice sends to Bob. Another direction is to consider FL in networks where packets can travel simultaneously on *multiple paths*, as in the SMT framework [10]. Recently, Amir *et al.* [2] presented a protocol for this setting, that optimizes for low *communication overhead*. Designing such protocols that optimize for low *storage and computational overhead* remains an interesting open question.

Acknowledgements. We thank Jennifer Rexford and Eran Tromer for very useful discussions and collaborations on failure localization and routing security in general. Shai Halevi and the anonymous EuroCrypt reviewers made a number of very helpful comments to improve the presentation of this paper. Boaz Barak is supported by NSF grants CNS-0627526 and CCF-0426582, US-Israel BSF grant 2004288 and Packard and Sloan fellowships. Sharon Goldberg is supported by NSF grant CNS-0627526. David Xiao is supported by a NSF Graduate Research Fellowship and a NDSEG Graduate Fellowship, and part of this work was done while visiting the Institute for Pure and Applied Mathematics at UCLA.

References

[1] Traceroute. *Available: http://costard.lbl.gov/cgi-bin/man/man2html?traceroute+8*, 2001.

[2] Y. Amir, P. Bunn, and R. Ostrovsky. Authenticated adversarial routing. In *Theory of Cryptography Conference, TCC*, 2009.

[3] K. Argyraki, P. Maniatis, O. Irzak, A. Subramanian, and S. Shenker. Loss and delay accountability for the Internet. *ICNP*, 2007.

[4] I. Avramopoulos, H. Kobayashi, R. Wang, and A. Krishnamurthy. Highly secure and efficient routing. In *IEEE INFOCOM*, 2004.

[5] B. Awerbuch, D. Holmer, C. Nita-Rotaru, and H. Rubens. An on-demand secure routing protocol resilient to byzantine failures. In *ACM WiSE*, 2002.

[6] B. Barak, S. Goldberg, and D. Xiao. Protocols and lower bounds for failure localization in the Internet. In *IACR EUROCRYPT*, 2008.

[7] J. L. Carter and M. N. Wegman. Universal classes of hash functions. *JCSS*, 18(2):143–154, 1979.

[8] D. Clark, J. Wroclawski, K. Sollins, and R. Braden. Tussle in cyberspace: defining tomorrow's Internet. *IEEE/ACM Transactions on Networking*, 13(5):462– 475, June 2005.

[9] M. Crovella and B. Krishnamurthy. *Internet Measurement*. Wiley, 2006.

[10] D. Dolev, C. Dwork, O. Waarts, and M. Yung. Perfectly secure message transmission. *J. of the ACM*, 40(1), 1993.

[11] N. G. Duffield and M. Grossglauser. Trajectory sampling for direct traffic observation. *IEEE/ACM Trans. Networking*, 9(3), 2001.

[12] S. Goldberg, D. Xiao, B. Barak, and J. Rexford. A cryptographic study of secure fault detection in the internet. *Princeton University, Department of Computer Science, Technical Report*, 2007.

[13] S. Goldberg, D. Xiao, E. Tromer, B. Barak, and J. Rexford. Path quality monitoring in the presence of adversaries. In *SIGMETRICS*, June 2008.

[14] O. Goldreich. *Foundations of Cryptography*. Cambridge University Press, 2007.

[15] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *J. of the ACM*, 33(4):210–217, 1986.

[16] J. Hastad, R. Impagliazzo, L. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. of Computing*, 1999.

[17] J. He and J. Rexford. Towards Internet-wide multipath routing. *IEEE Network Magazine Special Issue on Scalablity*, March 2008.

[18] R. Impagliazzo and M. Luby. One-way functions are essential for complexity based cryptography. *FOCS*, 1989.

[19] R. Impagliazzo and S. Rudich. Limits on the provable consequences of one-way permutations. In *STOC*, 1989.

[20] V. Jacobson. Congestion avoidance and control. *ACM SIGCOMM*, 18(4), 1988.

[21] S. Kent, C. Lynn, and K. Seo. Secure border gateway protocol (S-BGP). *J. Selected Areas in Communications*, 18(4):582–592, April 2000.

[22] P. Laskowski and J. Chuang. Network monitors and contracting systems: competition and innovation. In *ACM SIGCOMM*, 2006.

[23] R. Mahajan, N. Spring, D. Wetherall, and T. Anderson. User-level Internet path diagnosis. *SOSP*, 2003.

[24] D. Mills, A. Thyagarajan, and B. Huffman. Internet timekeeping around the globe. *Proc. PTTI*, pages 365–371, 1997.

[25] A. T. Mizrak, Y.-C. Cheng, K. Marzullo, and S. Savage. Detecting and isolating malicious routers. *IEEE Transactions on Dependable and Secure Computing*, 3(3):230–244, 2006.

[26] M. Naor and G. N. Rothblum. Learning to impersonate. In *International Conf. on Machine Learning*, 2006.

[27] V. Padmanabhan and D. Simon. Secure traceroute to detect faulty or malicious routing. *HotNets-I*, 2002.

[28] L. Subramanian, V. Roth, I. Stoica, S. Shenker, and R. H. Katz. Listen and Whisper: Security mechanisms for BGP. In *NSDI*, 2004.

[29] E. L. Wong, P. Balasubramanian, L. Alvisi, M. G. Gouda, and V. Shmatikov. Truth in advertising: Lightweight verification of route integrity. In *PODC*, 2007.

[30] X. Zhang, A. Jain, and A. Perrig. Packet-dropping adversary identification for data plane security. In *CoNEXT: Conference on emerging Networking EXperiments and Technologies*, December 2008.

A Vulnerabilities of Other FL Protocols

We sketch why the protocols of [27, 5, 3] do not satisfy our security definition.

An On-demand Secure Routing Protocol Resilient to Byzantine Failures [5]: Awerbuch, Holmer, Nita-Rotaru and Rubens present a statistical FL protocol in which Alice and Bob run a secure *failure detection* protocol, where Bob sends out authenticated acks for each packet he receives. Once the number of detected faulty exchanges exceeds some threshold, say β, then Alice appends a encrypted list of "probed nodes" to each *new* packet that she sends out. If a node is included in the list of probed nodes, it is expected to send Alice an ack when it receives the packet containing the list. The acks are (basically) formed as in our "onion reports". To localize failures, Alice chooses probed nodes according to a binary search algorithm, until she localizes a single link.

Now, consider an adversary Eve that sits at R_i and, for every sent packet where R_i is not included in the list of probed nodes, Eve happily causes failures. Eve stops causing failures whenever R_i is included in the list of probed nodes. Alice will never be able to localize such an Eve to a single link; as long as Eve behaves herself when she is part of the list of probed nodes,

Alice has no way to find her. Our protocols avoid this problem by running their "detection phases" and "localization phases" on the same set of packets.

Furthermore, care must be taken in implementing this protocol in the presence of both adversarial behaviour and benign congestion. To see why, suppose that Eve causes the protocol to enter the localization phase. In [5], the binary search algorithm proceeds by one step each time failures are detected. It is important to ensure that normal congestion (on a link that is not adjacent to Eve) cannot cause the binary search algorithm to search for Eve in the wrong part of the path. To do this, the binary search algorithm should proceed by one step only when the *failure rate* exceeds some carefully chosen false alarm threshold (related to loss rate caused by normal congestion and the length of the portion of path that is currently being searched).

Packet Obituaries [3]: Argyraki, Maniatis, Cheriton, and Shenker propose an FL protocol that is similar to our Optimistic Protocol of Section 3.1. Each node locally stores digests of the packets they see, and at the end of some time interval, nodes send out reports to Alice that contain these packet digests. Alice then uses the information from these reports to localize failures on the path. The designers of this protocol focused on the benign setting, but mentioned that reports should also be *individually authenticated*. However, because these reports are not formed in a onion manner (as in our Optimistic Protocol) an adversarial node can implicate a innocent downstream node by selectively dropping the innocent node's reports.

Secure Traceroute [27]: At a very high level, Padmanabhan and Simon's FL protocol uses an approach that is very similar to that of [5]; Alice runs a failure detection protocol with Bob until she detects that more than a β fraction of her packets have been dropped. Then, on *subsequent (new) sent packets*, Alice will run a failure localization protocol, where the intermediate nodes are required to send out acks that are used to localize failures. However, this protocol is vulnerable to the same adversary as [5]'s protocol: an Eve that causes failures when Alice runs failure detection, and then behaves herself once Alice turns on failure localization. The other issue with this protocol is that acks are *individually authenticated*, rather than onionized in the localization phase.

B A Composition Technique for Statistical FL

We prove Lemma 3.3, Lemma 3.4, Lemma 3.6 and Lemma 3.8.

B.1 Proof of Lemma 3.3

Lemma B.1 (Restatement of Lemma 3.3)**.** *As long as* $T = O(\frac{K^2}{p(\beta-\alpha)^2} \ln \frac{K}{\delta})$*, then the estimators in the composition of SSS satisfy: for each* $i \notin E$ *where* E *is the set of nodes corrupted by Eve it holds (up to negligible error) that*

$$\Pr\left[\left|V_i - \tfrac{1}{T}(D_i + \tfrac{p'}{p}C_i)\right| > \tfrac{1}{4}\gamma\right] < \tfrac{\delta}{4(K+1)}$$

where $\gamma = \frac{\beta-\alpha}{2(K+1)}$*,* V_i *is* R_i*'s estimate of the failure rate between* i *and* $K+1$*,* D_i *is the number of data packets dropped between* R_i *and* R_{K+1}*, and* C_i *is the number of acks (destined for any node) dropped between* R_i *and* R_{K+1}*.*

Proof. Consider the random variable V_i' which is generated as V_i is generated in SSS, except that now we assume that instead using a pseudorandom function f_{k_1} to decide if a packet is a probe, as in equation (1), Alice and Bob instead use shared, truly random function ϕ_i. Now consider the statistical FL protocol composed of K instances of this "truly random version of SSS". In this statistical FL protocol, it follows that regardless of how Eve (or congestion) behaves,

- Every packet that Eve (or congestion) drops is a probe for each R_i with probability p independent of Eve's actions and each other R_j for $j \neq i$.

- Every ack that Eve (or congestion) drops or tampers with is a probe for each R_i with probability p' independent of Eve's actions and each other R_j for $j \neq i$. As we argued in the proof of Theorem 3.2, this is because acks intended for different nodes are indistinguishable (since they are all identically onion MAC'd, and they all originate at Bob), and since the acks are onion MAC'd, Eve cannot selectively tamper with the ack intended for an upstream node.

We will show now that this means V_i' is the average of many independent random variables. Let $S_{D_i}, S_{C_i} \subseteq [T]$ denote the set of exchanges that are data-faulty and ack-faulty, respectively for node R_i (we can order the exchanges in an arbitrary way), and notice that $S_{D_i} \cap S_{C_i} = \varnothing$ since an exchange cannot be both data- and ack-faulty. We now define the random variables X_t for $t \in [T]$. For $t \notin S_{D_i} \cup S_{C_i}$, the variable X_t is identically 0. Otherwise, X_t is defined as follows:

$$\text{For } t \in S_{D_i},\ X_t = \begin{cases} 1 & \text{w.p. } p \\ 0 & \text{w.p. } 1-p \end{cases}$$

$$\text{For } t \in S_{C_i},\ X_t = \begin{cases} 1 & \text{w.p. } p' \\ 0 & \text{w.p. } 1-p' \end{cases}$$

We claim that $V_i' = \frac{1}{pT}\sum_{t=1}^{T} X_t$ because each data packet is unique and sampled exchanges are chosen using a truly random function, each exchange will be sampled by R_i with independent probability p. Thus each data-faulty exchange in S_{D_i} was sampled by R_i with probability p, while each ack-faulty exchange in S_{C_i} was sampled by R_i with probability p' (because here we need to condition on the fact that at least one node sampled the ack-faulty exchange (so that Bob generates an ack for that exchange!)). Thus, it follows that $\mathbb{E}[V_i'] = \frac{1}{T}(D_i + \frac{p'}{p}C_i)$. We now have

$$\begin{aligned}
\Pr[|V_i' - \frac{1}{T}(D_i + \frac{p'}{p}C_i)| > \gamma] &= \Pr[|pV_i' - \frac{1}{T}(pD_i + p'C_i)| > p\gamma] \\
&= \Pr[\frac{1}{T}|\sum_t X_t - (pD_i + p'C_i)| > p\gamma] \\
&= \Pr[|\frac{1}{T}\sum_t X_t - \mu| > \frac{p\gamma}{\mu}\mu] \qquad (9)
\end{aligned}$$

where we let $\mu = \frac{1}{T}(pD_i + p'C_i)$. At this point, it would be nice if we could say that $\mu = O(p)$, which would allow us to derive the conclusion, but *a priori* we can only assume that $\mu = O(p')$. Instead, in the rest of this proof we shall carefully show that the probability that $C_i > 2\frac{p}{p'}T$ is at most $\frac{\delta}{8(K+1)}$, and conditioned on $C_i \leq 2\frac{p}{p'}T$ then we also have that the probability that $|\frac{1}{T}\sum_t X_t - \mu| > p\gamma$ is bounded by $\frac{\delta}{8(K+1)}$, which gives us an overall bound of $\frac{\delta}{4(K+1)}$.

We start by letting Y denote the number of exchanges in the game that require acks for any R_i. Notice that $\mathbb{E}[Y] = \frac{p}{p'}T$ and that $C_i \leq Y$ unconditionally, simply because one can't tamper with an ack if it was never sent. Now we split the probability in Equation (9) as follows:

$$\begin{aligned}
\Pr[|\frac{1}{T}\sum_t X_t - \mu| > \frac{p\gamma}{\mu}\mu] &\leq \Pr[|\frac{1}{T}\sum_t X_t - \mu| > \frac{p\gamma}{\mu}\mu \text{ and } C_i > 2\frac{p}{p'}T] \\
&\quad + \Pr[|\frac{1}{T}\sum_t X_t - \mu| > \frac{p\gamma}{\mu}\mu \text{ and } C_i \leq 2\frac{p}{p'}T] \\
&\leq \Pr[C_i > 2\frac{p}{p'}T] + \Pr[|\frac{1}{T}\sum_t X_t - \mu| > \frac{p\gamma}{\mu}\mu \ \mid\ C_i \leq 2\frac{p}{p'}T] \\
&\leq \Pr[Y > 2\frac{p}{p'}T] + \Pr[|\frac{1}{T}\sum_t X_t - \mu| > \frac{p\gamma}{\mu}\mu \ \mid\ C_i \leq 2\frac{p}{p'}T] \qquad (10)
\end{aligned}$$

Now by a Chernoff bound the first probability is much less than $\frac{\delta}{8(K+1)}$ for our choice of $T = O(\frac{1}{\gamma^2 p} \ln \frac{K}{\delta})$, since Y is the sum of independent $\frac{p}{p'}$-biased random variables.

The second probability can now be bounded by a Chernoff bound. Notice that while the *definition* of the distribution of the X_t depends on C_i, the actual randomness of the X_t is independent of C_i. This gives us that the second probability is bounded by

$$\Pr[|\frac{1}{T}\sum_t X_t - \mu| > \frac{p\gamma}{\mu}\mu \;\mid\; C_i \leq 2\frac{p}{p'}T] \leq 2^{-\Omega(\frac{p^2\gamma^2}{\mu}T)} \tag{11}$$

In the above, we are interested in the event $|\frac{1}{T}\sum_t X_t - \mu| > \frac{p\gamma}{\mu}\mu$ conditioned on $C_i \leq 2\frac{p}{p'}T$, so that we can bound

$$\mu = \frac{1}{T}(pD_i + p'C_i) \leq \frac{1}{T}(pD_i + 2pT) \leq 3p$$

Plugging this into Inequality (11) and recalling from the statement of Theorem 3.2 that we set $T = O(\frac{1}{p\gamma^2}\ln(K/\delta))$, we get

$$\Pr[|\frac{1}{T}\sum_t X_t - \mu| > \frac{p\gamma}{\mu}\mu \;\mid\; C_i \leq 2pT] \leq 2^{-\Omega(p\gamma^2 T)} \leq \frac{\delta}{8(K+1)}$$

Combining this with the previous bound on $\Pr[Y > 2\frac{p}{p'}T]$ gives us that Inequality (10) becomes

$$\Pr[|V_i' - \frac{1}{T}(D_i + \frac{1}{p}C_i)| > \gamma] \leq \frac{\delta}{4(K+1)}$$

To complete the proof of the lemma, it suffices to observe that if replacing a truly random function ϕ with a PRF alters the probability by more than $K\varepsilon_{\mathsf{prf}}$, then we can efficiently distinguish between ϕ and the PRF f with advantage $\varepsilon_{\mathsf{prf}}$ by using the distinguisher that simply simulates this entire game, using access to an oracle containing either ϕ or f to answer calls to the PRF from the scheme, and then outputting 1 iff the condition $|V_i - \frac{1}{T}(D_i + \frac{p'}{p}C_i)| \leq \frac{1}{2}\gamma$ is violated.[11] □ □

B.2 Proof of Lemma 3.4

Lemma B.2 (Restatement of Lemma 3.4)**.** *As long as $T = O(\frac{K^2}{p(\beta-\alpha)^2}\ln\frac{K}{\delta})$, for each $i, i+1 \notin E$ where E is the set of nodes corrupted by Eve it holds (up to negligible error) that*

$$\Pr\left[\frac{p'}{p}\frac{C_i - C_{i+1}}{T} > \frac{\gamma}{2}\right] < \frac{\delta}{2(K+1)}$$

where $\gamma = \frac{\beta-\alpha}{2(K+1)}$ and where C_i is the number of acks (destined for any node) dropped between R_i and R_{K+1}.

Proof of Lemma 3.4. Fix C_{i+1}. Let $M \leq T$ be the number of exchanges in the interval for which a data packet reaches Bob, and a corresponding ack packet returns to R_{i+1}. Since R_i is honest, $C_{i+1} - C_i$ will just be the number of acks that are dropped due to congestion on link $(i, i+1)$, which occurs with probability ρ. Let X_i be a ρ-biased $\{0,1\}$ variable.

Let $U = \frac{\gamma}{2}(1-(1-p)^K)T$. We can derive:

$$\begin{aligned}
\Pr[\tfrac{p'}{p}\tfrac{C_i - C_{i+1}}{T} > \tfrac{\gamma}{2}] &= \Pr[C_i - C_{i+1} > U] \\
&\leq \Pr\left[\sum_{j=1}^{M} X_i > U\right] = \Pr\left[\frac{1}{M}\sum_{j=1}^{M} X_i - \rho > (\frac{U}{\rho M} - 1)\rho\right] \\
&\leq 2^{-\Omega(\rho(\frac{U}{\rho M})^2 M)} = 2^{-\Omega(U^2/(\rho M))}
\end{aligned}$$

[11] This distinguisher in fact requires access to K oracles, either all computing either a truly random function or all computing a PRF. Then we can turn this into a distinguisher for a single oracle using the hybrid argument, which is why we lose a factor of K in the distinguishing advantage. See *e.g.,* [14] for details about this kind of argument.

Because we have $\gamma = \frac{\beta-\alpha}{2K} \gg \rho$, this implies that

$$\frac{U}{\rho M} \geq \frac{\frac{\gamma}{2}(1-(1-p)^K)}{\rho} = \Omega(1)$$

so $\Pr[C_i - C_{i+1} > U] \leq 2^{-\Omega(U)} = 2^{-\frac{\gamma}{2}(1-(1-p)^N)T}$. Substituting our value of T gives us that this is bounded by less than $\delta/(2(K+1))$. □ □

B.3 Proof of Lemma 3.6

Lemma B.3 (Restatement of Lemma 3.6)**.** *Let* $\Gamma = \frac{T}{K+1}\frac{\beta(2\alpha+\beta)}{\alpha+2\beta}$ *and* $\varepsilon_i = \frac{1}{2i}\frac{\beta-\alpha}{2\beta+\alpha}$*. For every* $i \in [K]$*, assume that* R_i *computes an estimate* V_i *that* $(\varepsilon_i, \delta_i)$*-estimates* $\|\mathbf{x}_i\|_p^p$*. Suppose also that* $\|\mathbf{x}_i\|_p^p \leq \frac{\beta i}{K+1}$*. Then with probability at least* $1-2\delta'$ *it follows that:*

1. *If "link* $(i, i+1)$ *is good" so that* $\|\mathbf{x}_{i+1}\|_p^p - \|\mathbf{x}_i\|_p^p \leq \frac{\alpha}{K+1}T$ *then* $V_{i+1} - V_i \leq \Gamma$*.*
2. *If "link* $(i, i+1)$ *is bad" so that* $\|\mathbf{x}_{i+1}\|_p^p - \|\mathbf{x}_i\|_p^p \geq \frac{\beta}{K+1}T$ *then* $V_{i+1} - V_i \geq \Gamma$*.*

Proof. We prove each case separately.

Link $(i, i+1)$ **is good.** Since V_i $(\varepsilon_i, \delta_i)$-approximates $\|\mathbf{x}_i\|_p^p$, we can apply (4) to find, that with probability $1-2\delta'$,

$$\begin{aligned}
V_{i+1} - V_i &\leq (1+\varepsilon_{i+1})\|\mathbf{x}_{i+1}\|_p^p + (1-\varepsilon_i)\|\mathbf{x}_i\|_p^p \\
&\leq (1+\varepsilon_{i+1})(\|\mathbf{x}_{i+1}\|_p^p - \|\mathbf{x}_i\|_p^p) + (\varepsilon_{i+1}+\varepsilon_i)\|\mathbf{x}_i\|_p^p \\
&\leq (1+\varepsilon_{i+1})\tfrac{\alpha}{K+1}T + (\varepsilon_{i+1}+\varepsilon_i)\tfrac{i\beta}{K+1}T \\
&= \tfrac{\alpha}{K+1}T\left(1+\varepsilon_{i+1}(1+\tfrac{\beta}{\alpha}i) + \varepsilon_i i\tfrac{\beta}{\alpha}\right) \\
&\leq \tfrac{\alpha}{K+1}T\left(1+(i+1)\varepsilon_{i+1}(1+\tfrac{\beta}{\alpha}) + i\varepsilon_i(1+\tfrac{\beta}{\alpha})\right) \\
&= \tfrac{T}{K+1}\tfrac{\beta(2\alpha+\beta)}{\alpha+2\beta} = \Gamma
\end{aligned} \tag{12}$$

where we get the required inequality by putting $\varepsilon_i = \frac{1}{2i}\frac{\beta-\alpha}{2\beta+\alpha}$.

Link $(i, i+1)$ **is bad.** Again, we apply (4) to find, that with probability $1-2\delta'$,

$$\begin{aligned}
V_{i+1} - V_i &\geq (1-\varepsilon_{i+1})(\|\mathbf{x}_{i+1}\|_p^p - \|\mathbf{x}_i\|_p^p) - (\varepsilon_{i+1}+\varepsilon_i)\|\mathbf{x}_i\|_p^p \\
&\geq (1-\varepsilon_{i+1})\tfrac{\beta}{K+1}T - (\varepsilon_{i+1}+\varepsilon_i)\tfrac{i\beta}{K+1}T \\
&= \tfrac{T}{K+1}\tfrac{\beta(2\alpha+\beta)}{\alpha+2\beta} = \Gamma
\end{aligned} \tag{13}$$

where we again get the required inequality by putting $\varepsilon_i = \frac{1}{2i}\frac{\beta-\alpha}{2\beta+\alpha}$. □

B.4 Proof of Lemma 3.8

Our proof of Lemma 3.8 relies on the assumption that Eve occupies less than $\sqrt{K}$ links on the path.

To better understand why we made this assumption, suppose Eve occupies a large number of links on the path, and let node R_e be node occupied by Eve. Suppose Eve adds a small number $\leq \frac{\beta}{K+1}$ of nonsense packets to each link she occupies that is upstream of node R_e. If the number of added packets at each link is small, then there is a probability greater than δ that Alice will not raise alarm for these links. Next, at node R_e, Eve drops all the packets she added before, and additionally causes γ-fraction of failures. As such, we have that V_e will be large (proportional to the number of nonsense packets added downstream), and V_{e+1} will be large as

well (proportional to the γT failures at R_e). It follows that there is a probability greater than δ that $V_{e+1} - V_e$ will be small enough for Alice not to alarm, and so security fails. To rule out this attack, we limit the number of nodes occupied by Eve; this forces Eve to add a larger number of nonsense packets to the links upstream of R_e and increases the probability that Alice will raise an alarm for one of these links.

However, our proof only uses a simple averaging argument to claim that if Eve occupies M links, there must be a *single* link where $\gamma T \geq \frac{\beta}{M} T$, and uses this to arrive at the fact that Eve can only occupy $M \leq \sqrt{K}$ links on the path. However, we have not used the fact that Eve must cause a total of βT failures at *all* the links she occupies; we conjecture that using this fact could allow us to arrive at a weaker bound on M. We leave this to future work.

Lemma B.4 (Restatement of Lemma 3.8)**.** *If Eve occupies $M \leq \sqrt{(K+1)(1 - \frac{\rho}{\beta} K^2)}$ links and causes a β-fraction of failures in the interval, then there must be a link $(i, i+1)$ that is adjacent to Eve with*

$$\|\mathbf{x}_{i+1}\|_p^p - \|\mathbf{x}_i\|_p^p \geq \tfrac{\beta}{K+1} T$$

Proof of Lemma 3.8. Since Eve occupies M links causes at least a β-fraction failures, it immediately follows that there exists a link $(i, i+1)$ adjacent to Eve where at least $\frac{\beta}{M}$-fraction of failures, *i.e.,* $D_{i+1} - D_i \geq \frac{\beta}{M}$. Now if the following holds

$$\|\mathbf{x}_{i+1}\|_p^p - \|\mathbf{x}_i\|_p^p > \tfrac{\beta}{K+1} T \tag{14}$$

we are done, since link $(i, i+1)$ is adjacent to Eve. Thus, suppose (14) do *not* hold. Then, applying identity (6), we have that

$$\begin{aligned} \tfrac{\beta}{K+1} T &\geq \|\mathbf{x}_{i+1}\|_p^p - \|\mathbf{x}_i\|_p^p \\ &= D_{i+1} - D_i + \|\mathbf{a}_{i+1}\|_p^p - \|\mathbf{a}_i\|_p^p \end{aligned}$$

rearranging and then using that fact that $D_{i+1} - D_i \geq \frac{\beta}{M}$ we get

$$\|\mathbf{a}_i\|_p^p \geq = \beta T (\tfrac{1}{M} - \tfrac{1}{K+1}) \tag{15}$$

Next, consider the next link $(j, j+1)$ that is occupied by Eve and is upstream of link $(i, i+1)$. Now again, if the following holds

$$\|\mathbf{x}_{j+1}\|_p^p - \|\mathbf{x}_j\|_p^p > \tfrac{\beta}{K+1} T \tag{16}$$

then we are done, since link $(j, j+1)$ is adjacent to Eve. So, we again suppose (16) does *not* hold. Since Eve does not occupy any links between R_{j+1} and R_i, and only congestion-related loss could have occurred on the links between R_{j+1} and R_i. It follows that $\|\mathbf{x}_{j+1}\|_p^p \geq \|\mathbf{x}_i\|_p^p + \rho(i - j - 1)$. Since (16) does *not* hold, we can apply identity (6) and the fact that $\|\mathbf{x}_{j+1}\|_p^p \geq \|\mathbf{x}_i\|_p^p + \rho(i-j-1) \geq \|\mathbf{a}_i\|_p^p + \rho(i-j-1)$ and the bound on $\|\mathbf{a}_i\|_p^p$ in (15) to get

$$\|\mathbf{x}_j\|_p^p > \beta T \left(\tfrac{1}{M} - \tfrac{2}{K+1} - \tfrac{\rho}{\beta}(i - j - 1) \right) \tag{17}$$

We continue this argument for all $m \leq M - 1$ links that are adjacent to Eve and upstream of link $(i, i+1)$. Finally, arriving at the last such link, which we call link $(e, e+1)$, we have

$$\begin{aligned} \|\mathbf{x}_{e+1}\|_p^p &> \beta T \left(\tfrac{1}{M} - \tfrac{m}{K+1} - \tfrac{\rho}{\beta}(i - e - 1) \right) \\ &> \beta T \left(\tfrac{1}{M} - \tfrac{M-1}{K+1} - \tfrac{\rho}{\beta} K \right) \end{aligned} \tag{18}$$

where the last inequality follows by putting $m \leq M-1$ and $i-e \leq K$. Now since by definition Eve does not occupy any links downstream of link $(e, e+1)$, we immediately have that $||\mathbf{x}_e||_p^p = 0$. It follows that link $(e, e+1)$ has

$$||\mathbf{x}_{e+1}||_p^p - ||\mathbf{x}_e||_p^p > \beta T(\tfrac{1}{M} - \tfrac{M-1}{K+1} - \tfrac{\rho}{\beta}K) > \tfrac{\beta}{K+1} \tag{19}$$

where the last inequality follows because we put $M \leq \sqrt{(K+1)(1-\frac{\rho}{\beta}K^2)}$. This concludes the proof of this lemma, since link $(e, e+1)$ is adjacent to Eve. □

C Lower Bounds

After introducing some notation and technical lemmata, we prove Lemma 4.4 and Lemma 4.5.

C.1 Technical Lemmata

Notation. For two random variables, X, Y, we denote their concatenation with either (X, Y) or with XY.

Statistical distance. Recall that we define the statistical distance between two random variables X, Y as $\Delta(X, Y) = \frac{1}{2}\sum_{x \in U} |\Pr[X = x] - \Pr[Y = x]|$ where U is the union of the supports of X and Y (for more background on statistical distance, see *e.g.,* [14]).

Lemma C.1. *For any random variables X, Y, Z, X' satisfying $X = \eta Y + (1-\eta)Z$ and $\Delta(X, X') \leq \varepsilon$, there exists random variables Y', Z' and $\eta' \in [\eta \pm \varepsilon]$ such that $X' = \eta' Y' + (1-\eta')Z'$ and $\Delta(Y, Y') \leq \frac{3\varepsilon}{2\eta}$.*

Proof. Define a randomized process F acting on the support of X, where for each $x \in \mathsf{supp}(X)$, $F(x) = 1$ with probability $p(x) = \frac{\eta \Pr[Y=x]}{\Pr[X=x]}$ and $F(x) = 0$ with probability $1 - p(x)$, and say $F(x) = 0$ for all $x \notin \mathsf{supp}(X)$. We can check that

$$\Pr[F(X) = 1] = \mathbb{E}[F(X)] = \sum_{x \in \mathsf{supp}(X)} \Pr[X = x] \tfrac{\eta \Pr[Y=x]}{\Pr[X=x]} = \sum_{x \in \mathsf{supp}(X)} \eta \Pr[Y = x] = \eta$$

and similarly $F(X) = 0$ with probability $1 - \eta$. Furthermore, we claim that $Y = (X \mid F(X) = 1)$ since for every x,

$$\Pr[X = x \mid F(X) = 1] = \tfrac{\Pr[F(X)=1 \wedge X=x]}{\Pr[F(X)=1]} = \tfrac{\Pr[F(x)=1]\Pr[X=x]}{\eta} = \tfrac{\eta \Pr[Y=x]}{\Pr[X=x]} \tfrac{\Pr[X=x]}{\eta} \Pr[Y = x]$$

and similarly $Z = (X \mid F(X) = 0)$.

Since $\Delta(X, X') \leq \varepsilon$, this means that $\Pr[F(X') = 1] = \eta'$ for $\eta' \in [\eta \pm \varepsilon]$, and also $\Delta((F(X), X), (F(X'), X')) \leq \varepsilon$. Define $Y' = (X' \mid F(X') = 1)$ and $Z' = (X' \mid F(X') = 0)$. We may derive:

$$\begin{aligned} \varepsilon &\geq \Delta((F(X), X), (F(X'), X')) \\ &= \Delta(\eta(1, Y) + (1-\eta)(0, Z), \eta'(1, Y') + (1-\eta')(0, Z')) \end{aligned}$$

Viewing the random variables as the characteristic vectors of their distributions, and using the ℓ_1 formulation of statistical distance, we have:

$$= \tfrac{1}{2}||\eta(1, Y) + (1-\eta)(0, Z) - \eta'(1, Y') - (1-\eta')(0, Z')||_1$$

Since coordinates of the form $(1, Y)$ are disjoint from coordinates of the form $(0, Z)$, we have the equality:

$$\begin{aligned}
&= \tfrac{1}{2}\|\eta(1,Y) - \eta'(1,Y')\|_1 + \tfrac{1}{2}\|(1-\eta)(0,Z) - (1-\eta')(0,Z')\|_1 \\
&\geq \tfrac{1}{2}\|\eta(1,Y) - \eta'(1,Y')\|_1 \\
&= \tfrac{1}{2}\|\eta Y - \eta' Y' - (\eta' - \eta)\|_1 \\
&\geq \tfrac{1}{2}\eta\|Y - Y'\|_1 - \tfrac{1}{2}|\eta' - \eta| \\
&\geq \eta\Delta(Y, Y') - \varepsilon/2
\end{aligned}$$

which, rearranged, gives us that $\Delta(Y, Y') \leq \frac{3\varepsilon}{2\eta}$. □

Lemma C.2. *Let X, Y, X', Y' be such that $\Delta(XY, X'Y') \leq \varepsilon$. Say that $x \in \mathsf{supp}(X) \cap \mathsf{supp}(X')$ is δ-*bad *if $\Delta(Y(x), Y'(x)) > \delta$, where $Y(x)$ denotes the conditional distribution $Y \mid X = x$ and $Y'(x)$ denotes $Y' \mid X' = x$. Then $\Pr[X$ is δ-bad$] \leq 2\varepsilon/\delta$.*

Proof. Our proof is by contradiction. Suppose $\Pr[X \text{ is } \delta\text{-bad}] > 2\varepsilon/\delta$. Then, use the triangle inequality to obtain:

$$\Delta(XY, X'Y') \geq \Delta(XY, XY'(X)) - \Delta(XY'(X), X'Y')$$

where the random variable $Y(X')$ denotes $Y(x) = y \mid x \leftarrow_{\mathrm{R}} X'$. By hypothesis, we know that $\Delta(X, X') \leq \varepsilon$ so we have

$$\begin{aligned}
&\geq \Delta(XY, XY'(X)) - \varepsilon \\
&= \Delta(Y, Y'(X)) - \varepsilon \\
&\geq \Pr[X \text{ is } \delta\text{-bad}]\Delta(Y \mid X\text{bad}, Y'(X) \mid X\text{bad}) - \varepsilon
\end{aligned}$$

and since the statistical distance between $Y(x)$ and $Y'(x)$ when x is δ-bad is at least δ, we have

$$> (2\varepsilon/\delta) \cdot \delta - \varepsilon \geq \varepsilon$$

which contradicts the hypothesis that $\Delta(X, X') \leq \varepsilon$. □

Lemma C.3. *Let X, Y, X', Y' be random variables where $\Delta(X, X') \leq \varepsilon_1$. We say that $x \in \mathsf{supp}(X) \cap \mathsf{supp}(X')$ is ε_2-*bad *if $\Delta(Y(x), Y'(x)) \geq \varepsilon_2$, and suppose $\Pr[X$ ε_2-bad$] \leq \varepsilon_3$. Then $\Delta(XY, X'Y')) \leq \varepsilon_1 + \varepsilon_2 + \varepsilon_3$.*

Proof. This follows from the triangle inequality:

$$\begin{aligned}
\Delta(XY, X'Y') &\leq \Delta(XY, XY'(X)) + \Delta(XY'(X), X'Y') \\
&\leq \Delta(Y, Y'(X)) + \Delta(X, X') \\
&\leq \Delta(Y, Y'(X)) + \varepsilon_1 \\
&\leq \Pr[X \ \varepsilon_2\text{-bad}] \cdot \Delta(Y \mid X \text{ bad},\ Y'(X) \mid X \text{ bad}) \\
&\quad + (1 - \Pr[X \ \varepsilon_2\text{-bad}]) \cdot \Delta(Y \mid X \text{ not bad},\ Y'(X) \mid X \text{ not bad}) + \varepsilon_1 \\
&\leq \varepsilon_3 \cdot 1 + (1 - 0) \cdot \varepsilon_2 + \varepsilon_1
\end{aligned}$$

□

C.2 Proof of Lemma 4.4

First a word about random oracles, which we treat as a function $\mathsf{RO} : \{0,1\}^* \to \{0,1\}$. We look at the RO using the "lazy evaluation" methodology: points in RO are not fixed until they have been queried. When an efficient algorithm executes with a random oracle, it can make only an efficient number of queries. This means that RO can be viewed as a polynomially long string representing the responses to the algorithm, rather than as an infinitely large function, and replacing RO by a different string RO' (of equal length) amounts to replacing the real random oracle with a "fake random oracle". Thus, in the following, when we say that an oracle outputs a fake random oracle consistent with the output h of an algorithm A, we mean it outputs a string $\mathsf{RO}' \in \{0,1\}^{\mathrm{poly}(n)}$ such that running A with the responses encoded in RO' generates h.

Learning Algorithm. We apply Naor and Rothblum's [26] learning algorithm for *adaptively changing distributions (ACD)*. The ACD we work with is defined as a pair (Init, D) of random processes, where Init is a key generation process that takes a uniform string s and generates secrets $\overrightarrow{\mathsf{aux}} = \mathsf{Init}(s)$, and D is a process that takes the initial state $\overrightarrow{\mathsf{aux}}$ and a history H_{i-1} and uses them to generate a sample τ_i of an exchange. The history H_{i-1} consists of tuples (r_j, τ_j) for all $j \leq i-1$, where r_j was the random string used to generate the transcript τ_j. Notice that the τ_j are the outputs of the ACD, while the initial state s and the r_j remain secret.

Theorem C.4 (Naor and Rothblum [26])**.** *There exists a* PSPACE *algorithm that, for any ACD* (Init, D)*, observes at most* $t = O(n/\varepsilon^4)$ *samples from* D *and generates with probability* $> 1-\varepsilon$ *a fake secret state* $\overrightarrow{\mathsf{aux}}'$ *and fake history* H'_t *such that simulating* D *with* $\overrightarrow{\mathsf{aux}}', H'_t$ *generates a sample* tau'_{t+1} *that is distributed* ε*-statistically close to an honest sample generated by* D *using* $\overrightarrow{\mathsf{aux}}, H_t$*.*

Lemma C.5 (Lemma 4.4 restated)**.** *Relative to a* $(\mathsf{PSPACE}, \mathsf{RO})$*-oracle, there exists an efficient algorithm that observes at most* $t = O(\frac{n}{\varepsilon^4})$ *honest* exchanges $\tau_j = \langle R_{i-1}, R_i \rangle_j$ *and then, with probability* $> 1-\varepsilon$*, outputs algorithms* $R'_0, \ldots, R'_{K+1}$ *such that a fake exchange* $\langle R'_{i-1}, R'_i \rangle_{t+1}$ *is distributed* ε*-close to an honest exchange* $\langle R_{i-1}, R_i \rangle_{t+1}$*.*

Proof of Lemma 4.4. Apply Theorem C.4 where the Init function is our key generation function, and D is the algorithm that simulates the interaction of all algorithms $R_0, \ldots, R_{K+1}$ given a uniformly random data packet to be sent, including simulating all the congestion along links between the nodes, and outputs the transcript along link $(i-1, i)$. To generate the transcript of the i'th exchange, D takes input $\overrightarrow{\mathsf{aux}}, H_{i-1}, \mathsf{RO}_i, r_i$ where RO_i are responses to new queries to the random oracle that D makes in generating the transcript, r_i is the fresh internal randomness used to generate the $i+1$'th transcript, and $H_{i-1} = (\tau_j, \mathsf{RO}_j, r_j)_{j \leq i-1}$ is a history of previous transcripts, responses of the random oracle, and internal randomness. Notice that because D simulates *all* the nodes, there is *no distinction* between how the learning algorithm treats RO_i and r_i.

After observing $t = O(n/\varepsilon^4)$ exchanges, using the learning algorithm of Theorem C.4, we get with probability $> 1-\varepsilon$ fake secrets $\overrightarrow{\mathsf{aux}}', H'_t$ consistent with the transcripts and such that generating the $t+1$'th transcript using the fake secrets is ε to generating the $t+1$'th transcript using the honest secrets. Set R'_i to be R_i but with the secrets in $\overrightarrow{\mathsf{aux}}', H'_t$ hardwired into the algorithm.

Efficiency is clear because we allow a PSPACE oracle and because the number of samples is $O(n/\varepsilon^4)$. □

C.3 Proof of Lemma 4.5

Lemma C.6 (Lemma 4.5 restated)**.** *Suppose that* $\Delta(\langle A, B \rangle, \langle A', B' \rangle) \leq \varepsilon$*, and there exist events* $E_1, \ldots, E_r$ *over the internal randomness of* A, B *such that (1)* $\forall j$*, conditioned on* E_j*, the first* j *messages from* B *to* A *are independent of* A*'s internal randomness, and (2)* $\Pr[E_j \mid E_{j-1}] \geq \rho$*.*

Set $\varepsilon = (\rho/10)^{4r}$, Then there exist $\eta \geq (\rho/2)^r$, and distributions over the transcripts Y, Z such that $\langle A, B' \rangle$ is a convex combination $\eta Y + (1-\eta)Z$ and

$$\Delta((Y, \mathsf{view}_A(Y)), (\langle A, B \rangle, \mathsf{view}_A(\langle A, B \rangle) \mid E_r)) \leq 1/100$$

Proof. We define $\sigma_i = \langle A, B \rangle^i$, the first i messages in the partial transcript of $\langle A, B \rangle$. Notice that σ_i is a random variable so it makes sense to condition σ_i on the event E_i. Similarly define $\sigma'_i = \langle A', B' \rangle^i$ and $\sigma_i^{\mathsf{alt}} = \langle A, B' \rangle^i$. We will decompose A, B, A', B' into next-message functions A_i, B_i, A'_i, B'_i for $1 \leq i \leq r$, where we assume that each party takes turns communicating, and $2r$ is the maximum number of rounds of communication. Recall from the statement of the lemma that we think of applying the i'th next message function $B'_i(\tau)$ to a partial transcript τ of $2i-1$ messages as sampling from $(\langle A', B' \rangle^{2i} \mid \langle A', B' \rangle^{2i-1} = \tau)$.

Recall that $\mathsf{view}_A(\tau)$ is the distribution of the internal randomness of A conditioned on outputting the transcript τ.

Now, we define the conditional view $\mathsf{condview}_A(\tau)$ to be a *uniformly sampled view* of all the possible shared and independent internal randomness for A that causes A to output its messages in τ, such that the shared randomness is consistent with B output its messages in τ. Intuitively, we think of $\mathsf{condview}_A(\tau)$ and randomness for A sampled under the assumption that parties A, B that created τ correctly shared their randomness.

Define alternating view $\mathsf{altview}_A(\tau)$ to be a uniformly sampled view of the all possible shared and independent internal randomness for A that causes A to output its messages in τ, *even if the shared randomness would not result in B outputting its messages in τ*. We can think of $\mathsf{altview}_A(\tau)$ and randomness for A sampled under the assumption that B and A use independent randomness, *i.e.,* that party B incorrectly shares its randomness with A.

Notice that when the parties that created τ are independent, the $\mathsf{condview}$ and $\mathsf{altview}$ are the same. Otherwise, the support of the $\mathsf{condview}$ is a subset of the support of the $\mathsf{altview}$. As such, we can list some properties of $\mathsf{condview}, \mathsf{altview}, \mathsf{view}$. Before we begin, recall that A, B share randomness, but B' impersonates to A, the randomness that B' is supposed to share with A is independent of A's randomness.

1. $\mathsf{condview}_A(\langle A, B \rangle) = \mathsf{view}_A(\langle A, B \rangle)$ because in this case both parties are correctly sharing randomness.

2. $\mathsf{altview}_A(\langle A, B' \rangle) = \mathsf{view}_A(\langle A, B' \rangle)$ because when B' is impersonating to A, B' is not correctly sharing randomness with A. Thus the view of A is independent of the randomness of B'.

3. $\mathsf{condview}_A(\langle A, B' \rangle) \neq \mathsf{view}_A(\langle A, B' \rangle)$ because $\mathsf{condview}_A$ assumes that A, B' correctly share randomness, but when B' impersonates to A this is not the case.

4. $\mathsf{altview}_A(\langle A, B' \rangle^{2i}) = \mathsf{altview}_A(\langle A, B' \rangle^{2i-1})$ since B' computes the $2i$'th message, this does not affect A's $\mathsf{altview}$ since $\mathsf{altview}_A$ is independent of the randomness used by B'.

5. Conditioned on E_i, $\mathsf{condview}_A(\langle A, B \rangle^{2i}) = \mathsf{condview}_A(\langle A, B \rangle^{2i-1})$ because E_i tells us that the $2i$'th message (computed by B) is independent of A's randomness. It follows that the $2i$'th message will not affect A's $\mathsf{condview}$.

6. Conditioned on E_i, $\mathsf{condview}_A(\langle A, B \rangle^{2i}) = \mathsf{altview}_A(\langle A, B \rangle^{2i})$ and $\mathsf{condview}_A(\langle A, B \rangle^{2i+1}) = \mathsf{altview}_A(\langle A, B \rangle^{2i+1})$. This follows because messages $2, 4, ..., 2i$ are computed by B and messages $1, 3, ..., 2i+1$ are computed by A, and E_i tells us that all the messages from B to A are independent of A's randomness.

The proof of this Lemma 4.5 rests on the following claim, which will prove by induction:

Claim C.7. *Assuming $\varepsilon = (\rho/10)^{4r}$, for each i, $0 \le i \le r$, there exist $\eta_i \ge \rho/2$ and random variables Y_i, Z_i such that*

$$\sigma_{2i}^{\mathsf{alt}} = \prod_{j=1}^{i} \eta_j Y_i + (1 - \prod_{j=1}^{i} \eta_j) Z_i$$

and, for $\delta_i = \sqrt{\varepsilon}(10/\rho)^i$,

$$\Delta((\sigma_{2i}, \mathsf{condview}_A(\sigma_{2i}) \mid E_i), (Y_i, \mathsf{altview}_A(Y_i))) \le \delta_i$$

Apply this claim for the case of $\sigma_{2r} = \langle A, B\rangle$ and $\sigma_{2r}^{\mathsf{alt}} = \langle A, B'\rangle$ to obtain $Y = Y_r$ and $\eta = \prod_{j=}^{r} \eta \ge (\rho/2)^r$. This implies that we have the decomposition $\sigma_{2r}^{\mathsf{alt}} = \eta Y + (1-\eta)Z$ Next, we argued above (in the first item) that $\mathsf{condview}_A(\sigma_{2r}) = \mathsf{view}_A(\sigma_{2r})$. We argued above (in the second item) that $\mathsf{altview}_A(\sigma_{2r}^{\mathsf{alt}}) = \mathsf{view}_A(\sigma_{2r}^{\mathsf{alt}})$, and the decomposition of $\sigma_{2r}^{\mathsf{alt}} = \eta Y + (1-\eta)Z$ then gives that that $\mathsf{altview}_A(Y) = \mathsf{view}_A(Y)$. Finally, we can apply the claim to obtain that

$$\Delta((Y, \mathsf{view}_A(Y)), (\sigma_{2r}, \mathsf{view}_A(\sigma_{2r}) \mid E_r)) \le \sqrt{\varepsilon}(10/\rho)^r$$

which proves the lemma since $\sqrt{\varepsilon}(10/\rho)^r = (\rho/10)^r \le 100$. □

We now prove Claim C.7.

Proof of Claim C.7. Our proof is by induction. The base case $i = 0$ is trivial. The inductive hypothesis for $i-1$ is as follows: There exists $\eta_{i-1} \ge \rho/2$, $\delta_i = \sqrt{\varepsilon}(10/\rho)^i$ and random variables Y_{i-1}, Z_{i-1} such that

$$\Delta((\sigma_{2i-2}, \mathsf{condview}_A(\sigma_{2i-2}) \mid E_{i-1}), (Y_{i-1}, \mathsf{altview}_A(Y_{i-1})) \le \delta_{i-1} \tag{20}$$

and

$$\sigma_{2i-2}^{\mathsf{alt}} = \prod_{j=1}^{i-1} \eta_j Y_{i-1} + (1 - \prod_{j=1}^{i-1} \eta_j) Z_{i-1} \tag{21}$$

Our claim will follow if we show that this is also the case for i.

We are ready to start proving the inductive step. First, we apply A_i to both terms in Inequality 20 and update the view to get (because this process is identical in both cases):

$$\Delta((\sigma_{2i-1}, \mathsf{condview}_A(\sigma_{2i-1}) \mid E_{i-1}), (\zeta_{2i-1}, \mathsf{altview}_A(\zeta_{2i-1}))) \le \delta_{i-1} \tag{22}$$

where for compactness we have set $\zeta_{2i-1} = Y_{i-1}A_i(Y_{i-1}, \mathsf{altview}_A(Y_{i-1}))$.

Applying Lemma C.2. Suppose that $\langle A, B\rangle$, $\langle A', B'\rangle$ that are statistically close, and consider a partial transcript σ_{2i-1} generated by A, B. Informally, we would like to show that it is extremely unlikely that the next message functions of B applied to σ_{2i-1} generates a transcript that is statistically far from the transcript generated by the next-message function of B' applied σ_{2i-1}. Formally, we do this by applying Lemma C.2, with

$$\begin{aligned} X &= (\sigma_{2i-1}, \mathsf{condview}_A(\sigma_{2i-1})), \qquad & Y &= B_i(\sigma_{2i-1}, \mathsf{condview}_A(\sigma_{2i-1})) \\ X' &= (\sigma'_{2i-1}, \mathsf{condview}_A(\sigma'_{2i-1})), \qquad & Y' &= B'_i(\sigma'_{2i-1}) \end{aligned}$$

Notice that that $(\sigma_{2i}, \mathsf{condview}_A(\sigma_{2i})) = XY$ and $(\sigma'_{2i}, \mathsf{condview}_A(\sigma'_{2i})) = X'Y'$, and we have $\forall i$,

$$\Delta(XY, X'Y') = \Delta((\sigma_{2i}, \mathsf{condview}_A(\sigma_{2i})), (\sigma'_{2i}, \mathsf{condview}_A(\sigma'_{2i}))) \le \Delta(\sigma_{2i}, \sigma'_{2i}) \le \Delta(\langle A, B\rangle, \langle A', B'\rangle) \le \varepsilon$$

where the last inequality follows from the hypothesis in Lemma 4.5. Next, we say that a fixed transcript and view $x = (\tau_{2i-1}, \mathsf{condview}_A(\tau_{2i-1}))$ is $2\sqrt{\varepsilon}$-bad if

$$\Delta(Y(x), Y'(x)) = \Delta(B_i(\tau_{2i-1}, \mathsf{condview}_A(\tau_{2i-1})), \quad B'_i(\tau_{2i-1})) > 2\sqrt{\varepsilon}$$

We can now apply Lemma C.2 to find that the probability that $(\sigma_{2i-1}, \mathsf{condview}_A(\sigma_{2i-1}))$ is $2\sqrt{\varepsilon}$-bad is at most $\frac{2\varepsilon}{2\sqrt{\varepsilon}} = \sqrt{\varepsilon}$. Before we move on, also observe that by the hypothesis in Lemma 4.5 we know that $\Pr[E_{i-1}] = \rho^{i-1}$ so that

$$\Pr[(\sigma_{2i-1}, \mathsf{condview}_A(\sigma_{2i-1})) \text{ is } 2\sqrt{\varepsilon} \text{ bad} \mid E_{i-1}] \leq \sqrt{\varepsilon}/\rho^{i-1} \tag{23}$$

Applying Lemma C.3. Informally, we want to argue that, if $(\sigma_{2i-1} \mid E_{i-1})$ and ζ_{2i-1} along with their views are statistically close (Inequality 22), and if $(\sigma_{2i-1} \mid E_{i-1})$ is rarely bad (Inequality 23) it follows that transcripts $(\sigma_{2i} \mid E_{i-1})$ and ζ_{2i} along with their views are also statistically close. We will do this using Lemma C.3, setting

$$\begin{aligned} X &= (\sigma_{2i-1}, \mathsf{condview}_A(\sigma_{2i-1}) \mid E_{i-1}), & Y &= B_i(\sigma_{2i-1}, \mathsf{condview}_A(\sigma_{2i-1})) \mid E_{i-1} \\ X' &= (\zeta_{2i-1}, \mathsf{altview}_A(\zeta_{2i-1})), & Y' &= B'_i(\zeta_{2i-1}) \end{aligned}$$

Notice that Inequality 22 tells us that $\Delta(X, X') \leq \delta_{i-1}$. Furthermore, we have that

$$XY \;=\; (\sigma_{2i}, \mathsf{condview}_A(\sigma_{2i-1}) \mid E_{i-1})$$

and setting $\zeta_{2i} \;=\; \zeta_{2i-1} B'_i(\zeta_{2i-1})$ we have that

$$X'Y' \;=\; (\zeta_{2i}, \mathsf{altview}_A(\zeta_{2i-1}))$$

Furthermore, from Inequality 23 it follows that $x \;=\; (\sigma_{2i-1}, \mathsf{condview}_A(\sigma_{2i-1}) \;\mid\; E_{i-1})$ is $2\sqrt{\varepsilon}$-bad (*i.e.*, $\Delta(Y(x), Y'(x)) \geq 2\sqrt{\varepsilon}$) with probability at most $\sqrt{\varepsilon}/\rho^{i-1}$. Thus, we can apply Lemma C.3 to obtain

$$\Delta(XY, X'Y') = \Delta((\sigma_{2i}, \mathsf{condview}_A(\sigma_{2i-1}) \mid E_{i-1}), (\zeta_{2i}, \mathsf{altview}_A(\zeta_{2i-1}))) \leq \delta_{i-1} + 2\sqrt{\varepsilon} + \sqrt{\varepsilon}/\rho^{i-1}$$

Before moving on, notice that this immediately implies that

$$\Delta((\sigma_{2i} \mid E_{i-1}), \zeta_{2i}) \leq \delta_{i-1} + 2\sqrt{\varepsilon} + \sqrt{\varepsilon}/\rho^{i-1} \doteq \gamma \tag{24}$$

Applying Lemma C.1: So far, we have been conditioning on E_{i-1}. We now condition on E_i. We want to say that because $(\sigma_{2i} \mid E_{i-1})$ and ζ_{2i} are close, and because $(E_i \mid E_{i-1})$ happens often, we can decompose ζ_{2i} so that part of it is close to $(\sigma_{2i} \mid E_i)$.

We shall do this using Lemma C.1. Set $X \;=\; (\sigma_{2i} \mid E_{i-1})$ and let $Y = X \mid E_i$ while $Z = X \mid \neg E_i$. By hypothesis in Lemma 4.5 the conditional event $(E_i \mid E_{i-1})$ occurs with probability ρ, so that $(\sigma_{2i} \mid E_{i-1}) = X = \rho Y + (1-\rho)Z$. From Inequality 24 we know that $\Delta(X, \zeta_{2i}) = \gamma$. We can now apply Lemma C.1 to find that there exist $\eta_i \in [\rho \pm \gamma]$ and random variables Y_i, Z_i such that

$$\zeta_{2i} = \eta_i Y_i + (1 - \eta_i) Z_i \tag{25}$$

and

$$\Delta((\sigma_{2i} \mid E_i), Y_i) \leq \tfrac{3\gamma}{2\eta_i} \tag{26}$$

Setting $\delta_{i-1} = \sqrt{\varepsilon}(10/\rho)^{i-1}$ and assuming $\varepsilon = (\rho/10)^{4r}$, since $n_i \geq \rho - \gamma$ implies that $\eta_i \geq \rho/2$, and Inequality 26 is bounded by $\delta_i = \sqrt{\varepsilon}(10/\rho)^i$. Applying $\mathsf{altview}_A$ to both terms in Inequality 26 we get

$$\Delta((\sigma_{2i}, \mathsf{altview}_A(\sigma_{2i}) \mid E_i), (Y_i, \mathsf{altview}_A(Y_i))) \leq \sqrt{\varepsilon}(10/\rho)^i$$

and then applying the fact that $\mathsf{condview}_A(\sigma_{2i}) = \mathsf{altview}_A(\sigma_{2i})$ conditioned on E_i, we get

$$\Delta((\sigma_{2i}, \mathsf{condview}_A(\sigma_{2i}) \mid E_i), (Y_i, \mathsf{altview}_A(Y_i))) \leq \sqrt{\varepsilon}(10/\rho)^i \tag{27}$$

which proves the first equation in our induction step (corresponding to Inequality 20).

Finally, we finish by proving the part of our induction step corresponding to Equation 21. We can derive

$$\sigma_{2i}^{\mathsf{alt}} = \sigma_{2i-2}^{\mathsf{alt}} A_{i-1}(\sigma_{2i-2}^{\mathsf{alt}}) B'_{i-1}(\sigma_{2i-2}^{\mathsf{alt}} A_{i-1}(\sigma_{2i-2}^{\mathsf{alt}}))$$

and using Equation 21, we get

$$\begin{aligned}
&= \prod_{j=1}^{i-1} \eta_j Y_{i-1} A_{i-1}(Y_{i-1}) B'_{i-1}(Y_{i-1} A_{i-1}(Y_{i-1})) + (1 - \prod_{j=1}^{i-1} \eta_j) \dots \\
&= \prod_{j=1}^{i-1} \eta_j \zeta_{2i} + (1 - \prod_{j=1}^{i-1} \eta_j) \dots
\end{aligned}$$

now we apply Equation 25 to get

$$\begin{aligned}
&= \prod_{j=1}^{i} \eta_j Y_i + \prod_{j=1}^{i-1} \eta_j (1 - \eta_i) Z_i + (1 - \prod_{j=1}^{i-1} \eta_j) \dots \\
&= \prod_{j=1}^{i} \eta_j Y_i + (1 - \prod_{j=1}^{i} \eta_j) W_i \qquad (28)
\end{aligned}$$

where we used "..." to represent the rest of the convex combination of the Y_j, Z_j's which we finally collected in a new variable W_i[12]. Recalling that $\eta_i \geq \rho/2$ for all i, we have that $\prod_{j=1}^{i} \eta_j \geq (\rho/2)^i = 1/\text{poly}(n)$.

Thus, combining Inequality 27 with Equation 28 completes the proof of our induction step. □

[12]We can do this because Z_i is from the decomposition of ζ_{2i} while W_i corresponds to the decomposition of $\sigma_{2i}^{\mathsf{alt}}$.