Catshadow Mix Network Threat Model

David Stainton

***Abstract*—We show that the catshadow decryption mix network messaging system provides a stronger threat model than even Signal used with Tor. In particular we show that decryption mix networks must deliver messages in at least two route segments where one of them is not controlled by the sender, in order to provide the security properties suitable for high risk users such as whistleblowers and journalists.**

I. Introduction

I herein shall describe a mixnet messaging system which endeavors to not only provide users with usual cryptographic assurances such as message confidentiality, integrity and, authenticity... but we have the additional goals of reducing the amount of metadata leaked to passive network observers and network operators. In cryptographic messaging systems like Signal and Wire, users leak their entire social graph to the operators of the network. Additionally, other important kinds of metadata are leaked such as message send time, message receive time, message size. All of these are available to the system operators without breaking any of the cryptographic systems used.

Catshadow was inspired by the Loopix paper [1] but differs in the privacy notions, threat model and message delivery scheme. The Loopix messaging system does not hide contact network locations from one another but instead message delivery is done using only forward Sphinx packets and therefore the sender knows the final destination.

In the usual designs described in much of the mix network literature, decryption mix networks function as end to end mix networks where the sending client knows the final destination of the message and knows what its ciphertext will look like. In contrast to this design, I shall describe a decryption mix network messaging system which prevents contacts from learning one another's network location and prevents clients from being able to predict the final ciphertext retreived by the receiving client.

II. Privacy Goals

Informally our main goal is to prevent network operators and passive network observers from learning who is communicating with whom. We achieve this privacy by means of the follow privacy notions as described in [2]:

- Sender Unobservability
- Receiver Unobservability
- Sender Receiver Unlinkability

This design retains the notion of Sender Receiver Unlinkability even with an adversarial sender.

This analysis is only applicable for online users since offline users are clearly not sending any messages. Furthermore, Sender Unobservability is achieved by means of clients sending decoy traffic so that network observers will not know when a legitimate message is sent.

III. Adversarial Model

Mixnets provide the Anytrust property which means that if a route consists of at least one honest mix then there is still uncertainty for a passive network observer for the correlation between sent and received messages. This of course is predicated on the assumption that the mixnet has sufficient traffic from a sufficient number of users. Conversely, if an adversary has compromised the entire route it is then defined to be a bad route and implies immediate correlation between sent and received messages.

Given that Katzenpost is designed for the asynchronous messaging use case, it's remarkable that queueing messages at the edge of the network also implies that classical intersection attacks with full granularity involve compromising one or more Providers whereas most of the mixnet literature on the topic assumes different situation where message flows to clients are visible by passive network observers. In Loopix and Katzenpost this is not the case because of the queueing and the traffic padded protocol used by clients to retreive messages.

The adversary needs to compromise the Providers in order to learn which client's message queue received a given message. Without this information, intersection attacks would take much longer because Providers can have many client message queues which would make passive network observations contain a low amount of statistical information. We can therefore say that even if the adversary compromises the sending Provider and the receiving Provider that this would only allow the adversary perform an intersection attack and would not immediately allow linking senders with the receivers if the mix network had enough users and enough traffic.

IV. Sphinx

The above goals are achieved by adding delay with a mix strategy and by some of the properties of the Sphinx cryptographic packet format [3] [4] where forward Sphinx packets are used along with Single Use Reply Blocks (SURB), an anonymous delivery token. The delivery of messages is due to the composition of two routes. The first route is selected by the sending client using a forward Sphinx packet. The second route is selected by the receiving client with a SURB reply Sphinx packet. In between these two routes the message dwells in a queue on one of the Providers. In other words: Alice sends a message to Bob's remote queue which is hosted by one of the

Providers. Sometime later Bob retreives the message from his remote queue by sending a forward Sphinx packet destined to his message queue service on the remote Provider. The payload of his Sphinx packet contains a valid query message AND a SURB which allows the Service to send an anonymous reply to the client.

The mix network is composed in a stratified topology [5] where the topology is published in the PKI network consensus document. These topological routing restrictions are enforced by the way in which mix servers use the Katzenpost cryptographic wire protocol. [6] That is to say, if a mix tries to violate the topological restrictions the cryptographic wire protocol authentication will fail thus preventing the connection from being made. This cryptographic wire protocol is also used for the interactions of the other Katzenpost mix network entities such as client and authority servers.

Clients compose Sphinx packets where the payload is end to end encrypted (albeit in a nested manner) to the destination Provider. However the destination Provider sends it's payload back to the client using the SURB, and this is end to end encrypted from the sending Provider to the client. Therefore the receiving Provider of this SURB reply only observes ciphertext because only the receiving client has the vector of keys which can decrypt the SURB reply payload.

V. Mix Network PKI

The mixnet PKI system [7] is the root of all anonymity and security dependencies in the system. Unlike Tor, Katzenpost does not allow mixes to join the network automatically. Instead each mix identity public key is whitelisted in the PKI configuration file to allow mixes to participate in the network. I will describe various attacks against the PKI in a proceeding section of this document. The Katzenpost PKI works very similarly to the Tor and Mixminion Directory Authority system.

Katzenpost [8], [9], [4] borrows some designs from The Loopix Anonymity System [1], such as the Sphinx cryptogaphic packet format [3], the Poisson mix strategy [10] and, the stratified topology [5]. However Katzenpost differs in a number of ways from Loopix.

VI. The Poisson Mix Strategy

Katzenpost uses a continuous time mix strategy called the Poisson mix strategy. As described in the Loopix paper, all clients use the same set tuning parameters to configure their Poisson processes which the client scheduler uses to determine delays between sending decoy and legit traffic: $\lambda_p, \lambda_l, \lambda_d, \mu$ Katzenpost uses it's PKI to distribute these tuning parameters to the mixes and clients. However, catshadow in it's current design does not use the λ_d parameter. If clients sent traffic other than loops there would be a need for drop decoy messages. The current design uses only loop decoy messages because clients only send queries which should result in a response sent via a SURB supplied by the client's query message. Possible modifications to this client scheduler design are discussed in the section on statistical disclosure attacks.

Clients also use a Poisson process to select the delay of each hop of the routes used by their forward Sphinx packets and SURB replies. The client therefore knows all the mix delays in the full round trip of the query and response. It is useful for the client to possess this round trip mix time information and we shall discuss it in a later section when we discuss achieving reliability via an Automatic Repeat reQuest protocol scheme. If tuned correctly a mix network using the Poisson mix strategy can maintain a given level of mix entropy by increasing or decreasing the mix delays in relation to frequency of sending decoy traffic. That is to say, increasing mix delays and increasing the frequency of decoy traffic both result in increased mix entropy. To drive the point home about anonymity gaurantees or lack thereof we can say that the Poisson mix strategy does not guarantee a particular mix entropy because when fewer users are using the mix network the entropy on all the mixes decreases. Furthermore the tuning and the resulting mix entropy is dependent on a myriad of factors besides the Loopix tuning parameters we mentioned for tuning the poisson processes. These additional factors include the number of users currently participating in the mix network, number of mixes, number of topology layers, number of Providers, distribution of users among the Providers etc.

Mix networks in general do not provide much protection against various types of traffic analysis if they do not have enough users using them. The Poisson mix strategy in particular makes an additional tradeoff against protecting users and does not guarentee a specific mix entropy but instead uses tuning parameters which are tuned to a specific range of participating users and for various other contraints that may not always be met. It is remarkable that the Mixminion mix network used the pool mix strategy which always guarantees egress messages in the mix are always selected from a pool that maintains a set threshold of messages where N number of messages are removed from a pool when the threshold is exceeded by N. This guarantees all your messages will be mixed with enough entropy assuming a high enough threshold is selected and there are enough users. However the pool mix makes this guarantee with a huge performance trade-off where message latency can be very high. This presents a useability problem that would likely be a turn off for many users. Can the Poisson mix strategy be improved such that it receives dynamic tuning based on the number of users? Or is there a different mix strategy which would be safer to use than the Poisson mix but higher performance than the pool mix strategy?

Katzenpost aims to be a general purpose secure messaging transport which can be used to compose various messaging systems for multiple applications. Katzenpost also has a particular usage of Sphinx SURBs that is not found anywhere else. Instead of allowing users to exchange SURBs with a Nymserver [11] to send each other messages where anonymous replies are possible, we instead elect to have clients send SURBs to mixnet services so that a query response and be routed to the client without exposing the client's network location.

Clients interact with services on the mix network which run on mixes we designate as Providers. These Providers occupy the first and last of the layers in the stratified network topology where all the mixes in a given layer are only allowed to send messages to the mixes in the next topology layer. In Katzenpost this topology is strictly enforced by the Noise transport protocol authentication [8] [6]. That is, each mix only allows inbound connections to send it packets and inbound connections are only allowed from the previous layer in the topology.

VII. Catshadow

The catshadow usage of the Katzenpost framework is rather simple and only involved customizing two components: a server plugin to run the remote message queue on the Providers and a custom mixnet client.

In catshadow, the retrieval of messages from the remote message queue uses a special feature of the Sphinx cryptographic packet format called Single Use Reply Blocks. SURBs are essentially a short lived delivery token allowing anonymous replies. In Katzenpost, SURBs are generated by clients and sent along with a query to a service running on a Provider. The service can then use the SURB to send it's query response. Catshadow clients use this simple strict query response SURB based protocol to talk to a remote message queue. This gives clients strong location hiding properties from one another.

All communication to and from participants in the Katzenpost Mix Network is done via the Katzenpost Mix Network Wire Protocol [6]. The client generates a link key pair and use it to connect to a randomly choosen Provider in the set of all Providers in the network. This information must first be gathered by querying the mixnet PKI for a consensus document. Providers allow clients to connect with any key however the keys will be garbage collected if unused for more than one hour. This allows for client disconnects and reconnects such that clients will be able to retreive messages queued locally on the Provider. This requires the client to randomize it's selection of Providers only after some threshold down time duration or if switching physical locations and thus access to the Internet.

Upon first initialization, clients connect to the mix network and then select a random Provider and create a remote spool on it using Sphinx SURBs to interact with the remote spool service plugin. Although the client's queries to the remote service are done with a forward Sphinx packet whose end to end encryption terminates on the destination Provider, the replies using the client generated SURB are encrypted using a key that only the client knows. Two clients, Alice and Bob after intializing their clients with remote queues they can initiate a PANDA exchange. They use PANDA to exchange not only key material but their queue location information. They each check their own queue for messages from other contacts. Once the keys are exchanged, a triple diffie hellman variant is used to initialize both client's double ratchets. Each client uses one double ratchet per contact and therefore performs trial decryption on all messages received from their remote message queue.

VIII. Denial of Spam

In general it is not possible to receive spam in this messaging system. However two clients may confirm they are speaking to the same contact if they share their contact queue location information. Queue location is designated to be a queue identity number and a Provider name string. A user can receive messages from unwanted clients however unless their has been a prior key exchange the messages will fail to decrypt.

IX. Identity Leakage to Clients

Clients that share contact information can determine the intersection of their contacts by comparing queue identity information.

X. Metadata Leakage to Providers

The Provider of a client's remote message queue doesn't immediately learn which client appended each message. However it is possible for an adversary in this position to perform long-term statistical disclosure attacks which correlate received messages to the sets of users online at the time the messages were received. This assumes the adversary has compromised the Provider and can watch the network traffic on the other Providers. See section on statistical disclosure attacks.

XI. Denial of Service

Catshadow does not prevent someone from flooding the mixnet with Sphinx packets. Although there is optional per client rate limiting on the Providers, an adversary wishing to denial-of-service the network can easily generate many client keys and initiate many connections to the Providers to increase their rate of packets. Furthermore, the SURB based query response protocol amplifies such attacks by a factor of two.

There are additional denial of service attacks, such as creating too many accounts on each Provider and, filling up the local and remote message queues. The Directory Authority PKI for Katzenpost [7] can also be made to deny service. In particular the current version does not use a byzantine fault tolerate protocol and can therefore be prevented from generating a consensus file in each voting round. This would effectively DOS the whole mix network because without a consensus file to distribute key material, the network cannot be used.

XII. Classical Mix Network Attacks

There are five major categories of classical decryption mix network attacks and they are:

1) n-1 attacks [12]
2) epistemic attacks
3) compulsion attacks [13]
4) tagging attacks
5) statistical disclosure attacks

A. n-1 Attacks

The poisson mix strategy has a very simple n-1 attack where the adversary drops or delays all input messages to a given mix but let's the target message enter once the mix is probably empty. The adversary simply waits until the mix routes the message to the next hop. This amounts to a sort of denial of service attack that results in being able to trace one packet through the network when the n-1 attack is repeated on each mix in the route until the entire route is learned.

Katzenpost mixes can optionally send loop decoy Sphinx packets, however no defensive reaction is trigged by the loss of these loop messages as has been previously described [1], [14]. We are forced to use an imperfect heuristic to decide when loss of mix loops implies intentional packet loss for the purpose of orchestrating an n-1 attack or if it's benign or performance related packet loss. We plan to implement this heuristic detection and partial defense in the future. As it stands now Katzenpost has no defense against n-1 attacks.

XIII. Epistemic Attacks

Epistemic attacks [15], [16], [17] can occur when the adversary learns some information about a client's unique view of the network. Client path selections can be fingerprinted and are easily recognizable if they use a specific subset of available mixes. Therefore to prevent this attack we designed our PKI [7] to distribute the exact same consensus document to each client for a given epoch time duration.

XIV. Statistical Disclosure Attacks

Statistical disclosure attacks work to some extent on all anonymous communication networks. Catshadow is designed to provide partial defense against long-term intersection attacks as well as sufficient defence against short-term timing correlation attacks.

The classical mix network literature has described intersection attacks in terms of a mix network where a passive network observer can watch individual clients receive messages. This assumption can be otherwise stated that the adversary observers all the inputs and outputs of the mix network and thus receives a high granularity of statistical information. However the Loopix design makes use of message queues on the edge of the network so that messages can be received asynchronously. It turns out that this design also reduces accuracy of the stastical information available to passive network observers who will only see a specific Provider received a Sphinx packet but not know which client queue received it. That is to say, the adversary must compromise the receiving Provider in order to determine which users receive a given message. However there will certainly be longer term statistical information leaked to passive adversaries, especially if user behavior is repetative and predictable.

Catshadow forms bidirectional communication channels by combining unidirectional communication channels from two clients. Each of these unidirectional channel is a message queue operated by one of the Providers on the mix network. Users select a random Provider to connect to and herein lies a dilemma. If cleints connect to all Providers but the one they use to host their message queue, then these connections will eventually leak which Provider they are using to a passive network observer who is able to observe all of a client's interactions with the mix network. A client who changes their location before connecting has some defense against this.

The other aspect of this dilemma is if we choose to allow clients to select any Provider to connect to even if it happens to be the Provider they use to host their message queue then that Provider can perform stastical disclosure attacks to clients with message queue queries. If this linkage is established then the Provider learns the clients network location, unless Tor is used. Katzenpost allows Providers to optionaly only advertise their network location as a Tor onion service. Tor onion service usage is not adequate location hiding defense for our adversarial model because of being broken in a mere few seconds by a sufficiently global adversary, however I mention it here to indicate the attack would then proceed to break Tor to find the client's location.

Clients retreive messages by sending SURBs bundled in Sphinx packets destined to the Provider hosting their message queue. In this manner clients must periodically poll for new messages. Catshadow currently uses the FIFO queue scheduler as described in [1]. This however leaks statistical information in that messages sent to all of the Providers are not uniformly distributed. A passive adversary who collects enough information will be able to determine which Provider corresponds to a specific client for hosting it's message queue.

One possible solution I have not yet implemented for catshadow would be to enforce uniform distribution of messages among all Providers. That is, the client would send decoy loops to each Provider in equal portions except that the Provider it's polling would receive somewhat fewer decoy messages. This would necessarily add more latency on the processing time of the client's egress FIFO queue. The latency would grow linearly with the number of Providers on the network. Another somewhat equivalent solution would be to use a separate Poisson process to generate delays for processing one egress FIFO queue per destination Provider.

XV. Tagging Attacks

The Sphinx cryptographic packet format [3] allows for a one bit tagging attack under certain circumstances. The reason for this security compromise is to allow for the design of the Single Use Reply Block. The Sphinx header is MAC'ed but the packet body is not. Instead, the body is encrypted with a wide-block cipher (an SPRP). This ensures that an expected tag in the beginning of the plaintext can be used to verify the plaintext of the final decryption. Therefore in order to make use of this to perform a tagging attack, the adversary must have access to the result of the final SURB reply decryption as well as the ability to tag the packet some number of hops earlier in the route.

Without compromising any client devices there does exist a tagging attack in catshadow: If the adversary has compromised the Provider that a given client is interacting with, a tagging attack can be used to confirm that a given Sphinx packet sent by the client is the same packet received by a specific Provider. By flipping one or more bits in the Sphinx packet body, the adversary ensures that the Sphinx decryption on the destination Provider will fail due to not finding the expected authentication tag in the plaintext. However this attack is made more difficult due to the link encryption between nodes in the Katzenpost mix network. [6] Without breaking the link crypto the adversary would have to compromise the client's Provider in order to flip a bit in the body of the client's Sphinx packet. Catshadow chooses Providers at random upon startup unless recently disconnected.

XVI. Compulsion Attacks

Compulsion attacks are a problem for decryption mix networks since the cryptographic transformation of the Sphinx packets are used to form our communication channels we do not have the forward secrecy properties using new key exchanges in both directions. Instead we achieve partial forward secrecy by mix key rotation and redistribution of mix keys via the mixnet PKI. [7] This effectively limits the use of a mix key if it is compromised.

The other defense used in Katzenpost is to use link encryption between components in the mix network. Our Noise based protocol [6] uses a post quantum hybrid forward secret handshake pattern to make it difficult for adversaries to capture a Sphinx packet for use in a compulsion attack.

The usage of SURBs make Katzenpost and catshadow vulnerable to compulsion attacks if the adversary compromises the Provider which receives a SURB from a client. In that case, the adversary would learn the first hop from the SURB and then compell that mix operator into disclosing their mix key. The adversary uses the mix key to decrypt the Sphinx header component of the SURB and then learns the next hop. The compulsion attack continues until the destination Provider and queue identity is discovered by the adversary. After performing this Sphinx packet compulsion attack the adversary must link the Provider and queue identity to a specific user in order to learn the receiving client's network location (e.g. home IPv4 address).

Although there are other defenses [13] we'd like to implement in the future, the primary defenses for this attack are Katzenpost Noise based PQ hybrid link transport protocol [6] and mix key rotation [7]. Another possible future defense are forward secret mixes [18] which use an alternate key once they decryption a client identity. This allows the mix to destroy the mix key immediately after use however it also leaks information to the mix about which entity is sending the packets. This security versus metadata leakage tradeoff should be carefully examined in terms of the overal system design to determine if it's a design worthy of inclusion. At this time there are no plans to include forward secure mixing in Katzenpost.

In should be clear that selecting organizing the topology into several MLATs and in geographically distant locations would in theory increase the difficulty in performing a compulsion attack using legal means. I expect that powerful groups such as the NSA would be willing to illegally compromise all the mixes in the network or all the mixes in a specific route they are interested in pursuing. To do this I assume that the NSA stockpiles zero-days for the common hardware and software run by some or all of the mixes and have completely automated using remote code execution vulnerabilities and escalating privileges with additional attack payloads etc. Therefore the operational security of the mixes and directory authorities are outside the scope the threat model described in this document. However the Katzenpost software project should try to implement some features that make additional security hardening easier. Katzenpost does not at this time support usage with a hardware security module. Mix keys are deleted some time after they expire and the current epoch duration is set the three hours.

XVII. Compulsion Attacks via Path Selection

Katzenpost uses the stratified topology and therefore each message a client sends uses a newly selected random path through the network. If the adversary has compromised a mix in each network topology layer then eventually a client will select a bad route. In the mix network adversary model a bad route is defined as a route in which each hop is compromised by the adversary. This allows the adversary to link input and output mix network messages. However catshadow communication channels between contacts always involve two full routes through the mix network. Therefore compromising a single route would not be sufficient to link two clients on the network.

The strategy used by [19] and [20], where a set of users uses cascade published by the PKI for some fixed time duration, may be a better probability tradeoff. On the other hand it may be a worse tradeoff for clients that happen to select a bad cascade. I have yet to see a probability analysis that justifies one topology strategy of the other.

XVIII. Confirmation Attacks

I shall describe an active confirmation attack against Sphinx based an end to end reliability protocol using an Automatic Repeat reQuest scheme. Currently the catshadow messaging system does not provide end to end reliability. I plan to add reliability and the described partial defense against the active confirmation attack in the future.

This protocol feature exposes catshadow clients to an active confirmation attacks by adversaries which have compromised a client's message queue Provider. In this attack the adversaries goal is find the Provider which the client is locally connected to. The adversary divides the set of all Providers into two group and prevents one of the groups from receiving the SURB reply from the queue service. In this attack the adversary sees the client sequence number in the message queue query and thus can learn which group the client was in from subsequent messages. This attack can be performed in $\log_2 N$ steps where N is the number of Providers in the network.

The partial defense against this attack is for clients to randomize the retransmission delays above some threshold where it is assumed the adversary will not be willing to cause outages for that length of time. However at this time we do not have even a heuristic defense against this attack designed or implemented. However it seems the mix decoy loops can be used by Providers to learn when they are potentially experiencing an outage as part of an active confirmation attack. However unlike the heartbeat protocol design defense against n-1 attacks mentioned earlier, in this case sending additional decoy messages obviously does prevent the attack. However if the Provider responded by denying service for some period of time this could delay or prevent the attack. Clients can also use their decoy loops to learn when there is a potential routing outage due to a potential active confirmation attack.

If I expand the catshadow messaging system to include a SURB based publish subscribe protocol where clients upload multiple SURBs to a Provider in order to receive future messages for the given subscription then this too will expose clients to an additional attack. Upon receiving multiple SURBs from the client, the adversarial Provider could immediately send many SURB replies using all of the SURBs while simultaneously observing the received message rate for each Provider. If the entropy introduced by the messages from other users is low enough and the number of received SURBs is high enough there is some non-negligible probability that all the adversary replies will arrive at the client's Provider within some probable window of time which would allow the adversary to gain a statistical confirmation of the client's Provider.

XIX. PANDA Attacks

An early prototype version of Catshadow used the PANDA protocol [21] to facilitate not only key exchange but the establishment of bidirectional communications channels between communication partners. Later PANDA was replaced with the Reunion protocol.

XX. Reunion: an improved asynchronous anonymous PAKE protocol

Reunion is better than PANDA for several reasons including not being vulnerable to precomputation attacks by the server, leaking far less metadata so we can actually hide the fact that a successful key exchange has even occured. Attacks on the Reunion protocol are outside the scope of this document.

XXI. Public Key Infrastructure Attacks

Katzenpost has a PKI implementation which uses a deterministic voting protocol between multiple security domains and is inspired by the Tor and Mixminion Directory Authority systems. The Katzenpost Directory Authority uses a non-byzantine crash fault tolerant deterministic voting protocol to generate the consensus document containing signatures from a threshold number of authority protocol participants. Each directory authority shares some configuration in that they each must possess identical whitelists of the public identity keys for each mix and authority in the mix network.

This current design can have each and every voting round sabotaged by one or more bad acting protocol participants. It is my understanding that using a modern byzantine fault tolerant protocol would solve this problem.

The more important security problem here is that if the adversary compromises K of N total protocol participants then they get to decide the contents of the mix network consensus document. Controlling the contents of this document can be used to only advertise mixes controlled by the adversary for example. Therefore a real world deployment of this system should make sure K is sufficiently high to at least make the cost higher. Again we should reiterate, the NSA and some other powerful organizations probably stockpile zero-days, they may indeed have fully automated exploiting various remote code execution vulnerabilities that can be used to compromise mixes and authorities in a public deployment of this mix network. It seems our only partial defense against these attacks are to make them more noticeable and thus increase the risk to the groups that would perform such attacks against an anonymous communications network.

There are additional anonymity considerations with the Katzenpost PKI. Katzenpost is a continuous time mix strategy meaning that messages do not progress along their route through the network is fixed protocol rounds. Therefore to faciliate smooth transition from one epoch to the next where mix keys are rotated, there must be some grace period where mixes perform trial decryption on received Sphinx packets to determine which key to use, the old mix key or the new epoch's mix key.

Likewise there is a grace period for topology rerandmization which leads to a temporary reduction in mix entropy. Mix placement in the topology is randomized using a shared secret as a seed. Each authority voting round also produces a new shared random value using a hash based commit and reveal protocol. This topology rerandomization only occurs when a threshold number of mixes per layer are removed from the consensus document because their descriptors were not uploaded. That is, mix outages are detected in the next voting round by the absence of their descriptors.

Rerandomising the topology is helpful when the layered topology becomes unbalanced because it can prevent a single mix operator from using their mix as the only mix in a given layer. On the other hand we shouod not rerandomize topology too often because it effectively splits the anonymity set into two on each mix during the grace period. Said another way, from the perspective of a passive adversary, each input message originates from only one topology layer under normal circumstances and therefore there is no distinguishing characteristics among the messages which means there is logically only one set of messages being mixed. However in the case where messages originate from two different topology layers, this essentially splits the anonymity set into two sets because the message source becomes a distinguishable characteristic of each message for the grace period duration in order to facilitate a smooth transition between epochs which is what is needed by a continuous time mix strategy. This prevents clients from needing to apply special cased scheduling around epoch boundaries.

An adversary could prevent mixes to upload their descriptors to the PKI and thereby prevent them from being included in the consensus document. Additionally if the adversaries managed to cause some key outages it may cause the topology to be rerandomized in the next voting round and that reduced entropy could facilitate another type of attack.

XXII. Conclusion

We have systematically examined the attack vulnerabilities and adversary model for the catshadow decryption mixnet messaging system where the classical mixnet attack categories are:

1. n-1 attacks 2. compulsion attacks 3. tagging attacks 4. epistemic attacks 5. statistical disclosure attacks

However we also discuss a potential active confirmation attack for reliable retransmission protocols (ARQ protocols) which are layered on top of mix networks. In addition to that we use an anonymous, asynchronous PAKE protocol for the initial key exchanges between communication partners. The ACN used does augment the PAKE protocol's threat model and this does contribute to the overall system design of the catshadow messaging system.

The most important conclusion from the threat model analysis is that decryption mix networks have a fundamental design disadvantage where the sender always knows the final destination of their messages. This design "feature" is in direct opposition to the goal of Sender Receiver Unlinkability with Adversarial Sender. Anonymous messaging systems designed for high risk use cases need to provide this and other privacy notions in order to strongly hide user locations from one another. The Catshadow messaging system overcomes this design limitation of the Sphinx cryptographic packet format by having the messages total path be composed of two routes, the first chosen by the sender and the second route chosen by the receiver via a SURB reply.

Although there is a severe computational penalty, randomized reencryption mix networks a different set of tradeoffs than our messaging system for the privacy notions we desire. In particular, reencryption mix networks by defaults always provide the privacy notion Sender Receiver Unlinkability with Adversarial Sender. I'd like to point out that although reencryption mixnets provide this privacy notion without the added complexity which catshadow has. It is also clear that using the Poisson mix strategy is essentially a commitment to achieving high performance network throughput even if it means degrading the anonymity guarantees to essentially having no protection beyond what onion routing ACN's like Tor and I2p have to offer. Therefore we can say that Catshadow has differing anonymity properties depending on the amount of traffic flowing through the network. Whereas reencryption mixnets are severely limited in their ability to scale up the traffic capacity, however their threat model is typically stronger and their anonymity properties do not vary like Katzenpost does because of using the Poisson mix strategy.

XXIII. Acknowledgment

I would like to thank Leif Ryge for his creative input in the design of this mix network messaging system.

References

[1] A. Piotrowska, J. Hayes, T. Elahi, S. Meiser, and G. Danezis, "The loopix anonymity system," *arXiv preprint arXiv:1703.00536*, 2017. [Online]. Available: https://arxiv.org/pdf/1703.00536

[2] C. Kuhn, M. Beck, S. Schiffner, E. Jorswieck, and T. Strufe, "On privacy notions in anonymous communication," *Proceedings on Privacy Enhancing Technologies*, vol. 2019, no. 2, April 2019.

[3] G. Danezis and I. Goldberg, "Sphinx: A compact and provably secure mix format," in *Proceedings of the 30th IEEE Symposium on Security and Privacy (S&P 2009)*, 2009, pp. 269–282.

[4] Y. Angel, G. Danezis, C. Diaz, A. Piotrowska, and D. Stainton. (2017) Sphinx mix network cryptographic packet format specification. [Online]. Available: https://github.com/katzenpost/docs/blob/master/specs/sphinx.rst

[5] C. Diaz, S. J. Murdoch, and C. Troncoso, "Impact of network topology on anonymity and overhead in low-latency anonymity networks," in *Proceedings of the 10th Privacy Enhancing Technologies Symposium (PETS 2010)*, July 2010.

[6] Y. Angel. (2017) Katzenpost mix network wire protocol specification. [Online]. Available: https://github.com/katzenpost/docs/blob/master/specs/wire-protocol.rst

[7] Y. Angel, A. Piotrowska, and D. Stainton. (2017) Katzenpost mix network public key infrastructure specification. [Online]. Available: https://github.com/katzenpost/docs/blob/master/specs/pki.rst

[8] Y. Angel, G. Danezis, C. Diaz, A. Piotrowska, and D. Stainton. (2017) Katzenpost mix network specification. [Online]. Available: https://github.com/Katzenpost/docs/blob/master/specs/mixnet.rst

[9] ——. (2017) Katzenpost mix network specification. [Online]. Available: https://github.com/Katzenpost/docs/blob/master/specs/mixnet.rst

[10] D. Kesdogan, J. Egner, and R. Büschkes, "Stop-and-go MIXes: Providing probabilistic anonymity in an open system," in *Proceedings of Information Hiding Workshop (IH 1998)*. Springer-Verlag, LNCS 1525, 1998.

[11] G. Danezis, R. Dingledine, and N. Mathewson, "Mixminion: Design of a Type III Anonymous Remailer Protocol," in *Proceedings of the 2003 IEEE Symposium on Security and Privacy*, May 2003, pp. 2–15.

[12] A. Serjantov, R. Dingledine, and P. Syverson, "From a trickle to a flood: Active attacks on several mix types," in *Proceedings of Information Hiding Workshop (IH 2002)*, F. Petitcolas, Ed. Springer-Verlag, LNCS 2578, October 2002.

[13] G. Danezis and J. Clulow, "Compulsion resistant anonymous communications," in *Proceedings of Information Hiding Workshop (IH 2005)*, June 2005, pp. 11–25.

[14] G. Danezis and L. Sassaman, "Heartbeat traffic to counter (n-1) attacks," in *Proceedings of the Workshop on Privacy in the Electronic Society (WPES 2003)*, October 2003.

[15] G. Danezis and R. Clayton, "Route fingerprinting in anonymous communications," in *Sixth IEEE International Conference on Peer-to-Peer Computing (P2P 2006), 2-4 October 2006, Cambridge, United Kingdom*, 2006, pp. 69–72. [Online]. Available: https://doi.org/10.1109/P2P.2006.33

[16] M. Gogolewski, M. Klonowski, and M. Kutylowski, "Local view attack on anonymous communication," in *Proceedings of ESORICS 2005*, September 2005.

[17] G. Danezis and P. Syverson, "Bridging and fingerprinting: Epistemic attacks on route selection," in *Proceedings of the Eighth International Symposium on Privacy Enhancing Technologies (PETS 2008)*, N. Borisov and I. Goldberg, Eds. Springer, July 2008, pp. 133–150.

[18] G. Danezis, "Forward secure mixes," in *Proceedings of 7th Nordic Workshop on Secure IT Systems*, J. Fisher-Hubner, Ed., November 2002, pp. 195–207.

[19] N. Gelernter, A. Herzberg, and H. Leibowitz, "Two cents for strong anonymity: The anonymous post-office protocol," Cryptology ePrint Archive, Report 2016/489, 2016. [Online]. Available: https://eprint.iacr.org/2016/489.pdf

[20] H. Leibowitz, A. Piotrowska, G. Danezis, and A. Herzberg, “No right to remain silent: Isolating malicious mixes,” Cryptology ePrint Archive, Report 2017/1000, October 2017.

[21] J. Appelbaum and another cypherpunk. (2014) Going dark: Phrase automated nym discovery authentication. [Online]. Available: https://github.com/agl/pond/blob/master/papers/panda/panda.tex

Appendix A
Bibtex Entry

Note that the following bibtex entry is in the IEEEtran bibtex style as described in a document called ”How to Use the IEEEtran BIBTEX Style”.

```
@online{CatshadowThreatModel,
  title = {Catshadow Threat Model},
  author = {David Stainton},
  url = {https://sphinx.rs/papers/catshadow_threat_model/},
  year = {2020}
}
```