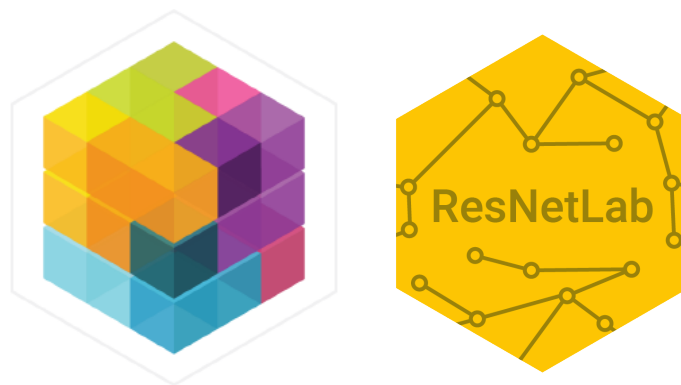




## Gossipsub-v1.1 Evaluation Report

Dimitris Vyzovitis, Yusef Napora, Dirk McCormick, David Dias, and Yiannis Psaras



Technical Report

**PL-TechRep-2020-001**

2020.04.18

©2020 Protocol Labs

This work is licensed under a Creative Commons Attribution 4.0 International License.

## CONTENTS

Contents	1
1 Introduction	1
2 Summary	2
3 Recommendations for users	4
4 Evaluation Setup	6
5 Bandwidth Requirements & Cost of Attack	8
6 Establishing the Baseline for Gossipsub	10
7 Eclipse Attack Against a Single Target	12
8 Eclipse Attack Against the Entire Network	15
9 Eclipse Attack Against all Publishers	19
10 Censor Attack Against a Single Target	23
11 Network Degradation Attack	27
12 Cold Boot Attack	37
13 Covert Flash Attack	45
14 Attack at Dawn	51
15 IP Collocation Attack	57

## Gossipsub-v1.1 Evaluation Report

DIMITRIS VYZOVITIS, YUSEF NAPORA, DIRK MCCORMICK, DAVID DIAS, and YIAN-NIS PSARAS, Protocol Labs

### 1 Introduction

**This document presents a comprehensive evaluation of version 1.1 of the Gossipsub protocol, as a Transaction and Block propagation protocol for general purpose blockchains.** As such, publishers in our setting are assumed to be the miners of the blockchain. The terms publishers and miners are therefore, used interchangeably throughout the report. Our evaluation targets the behaviour and responsiveness of Gossipsub in a non-adversarial environment (the Baseline Test Case), in order to see its latency and scalability bounds, and, most importantly, in adversarial environments under Sybil attacks. In the adversarial environment, Sybil nodes are trying to disrupt the network, either by completely disconnecting it or by withholding the truth from other nodes as they propagate messages. We have come a long way in terms of identifying different adversarial scenarios and we believe that we have covered all primitive attacks in the test cases we have considered.

Gossipsub-v1.1 is the hardened version of the protocol, which has been built on top of vanilla Gossipsub (Gossipsub-v1.0). In the remainder of this report, we refer to the hardened version of Gossipsub as Gossipsub-v1.1, or gs-v1.1. Similarly, we refer to the vanilla version of the protocol as Gossipsub-v1.0, or gs-v1.0. We have tested and are comparing the performance of gs-v1.1 against gs-v1.0.

This evaluation was done using the Golang implementation. You can find the code and the specs at:

- <https://github.com/libp2p/go-libp2p-pubsub>
- <https://github.com/libp2p/specs/blob/master/pubsub/gossipsub/gossipsub-v1.0.md>
- <https://github.com/libp2p/specs/blob/master/pubsub/gossipsub/gossipsub-v1.1.md>

## 2 Summary

### 2.1 Categories of Attacks

Here we provide a brief description of the main attacks we have considered and analyzed in the subsequent sections:

**Eclipse Attack:** Sybils target honest nodes and try to silence them and disconnect them from the network. This attack comes in three variants: the first variant targets a single victim, while more extreme variants target all publishers (i.e. miners) or even the entire network.

**Degradation Attack:** Sybils connect to the network and randomly drop messages with some probability. The objective is to degrade the performance of the network as a whole.

**Censorship Attack:** Sybils target a single node and attempt to silence it. At the same time, they behave correctly with regard to the rest of the network, so they propagate all messages except those from the target node.

**Cold Boot Attack:** In contrast to the above attacks, where Sybils enter a warmed-up network of honest nodes that have already formed their meshes and have built some score, in this attack both honest nodes and Sybils bootstrap together, with the Sybils attempting to eclipse the network and prevent it from booting.

**Covert Flash Attack:** Sybils enter the network together with honest nodes, as in the Cold Boot Attack., In contrast to the above case, they behave honestly for a period of time, which enables them to build a good score, then suddenly turn malicious and carry out an eclipse attack.

**Attack at Dawn:** Sybils enter the network before honest nodes and form a Sybil-only mesh. When honest nodes join shortly thereafter, they attempt to connect to a mesh dominated by Sybil nodes. Hence, the connection pool is poisoned, and honest nodes pick peers to form their initial mesh from a mix of honest and malicious nodes. In a resilient system, honest nodes should manage to recover the mesh by evicting misbehaving Sybil nodes.

We also evaluate the resistance of the protocol against a Sybil attack generated from a small pool of IP addresses. Although this is not an attack in itself, it allows us to place lower bounds to the cost of a Sybil attack. In particular, we show that gs-v1.1 has built-in protections against this case ( $P_6$  parameter), which forces the adversary to obtain a large number of IPs in order to perform an effective attack.

### 2.2 Mitigation Strategies

Gossipsub-v1.1 implements a number of different mitigation strategies to defend against the above attacks.

**Flood Publishing:** According to this mitigation strategy, publisher/miner nodes are advertising new messages (most commonly blocks) by sending the message to all of its connected peers, i.e. not only the ones in the node's mesh. Subsequent nodes do not flood the message, but operate according to their node status (e.g. within a mesh or through gossip). This mitigation strategy is not turning the protocol into a flooding protocol but only attempting to overcome situations where most (or even all) of the node's mesh connections are occupied by Sybils.

**Adaptive Gossip Dissemination:** Nodes emit gossip messages to a number of peers proportional to the number of peers they are connected to. The proportion is set by a parameter called the *gossip factor* and is set to 0.25 by default. Furthermore, there are three rounds of gossip in total, so every node has a probability equal to 0.57 of getting a message. Increasing the number of gossip messages a node sends proportionally to its connected peers overcomes the situation where most of the node's mesh connections are occupied by Sybil nodes and the network is also predominantly occupied by Sybils. The behaviour in gs-v1.0 was to gossip to a fixed number of nodes independently of the network size.

**Score Function:** The score function integrates a number of different weighted parameters with each one

focusing on rewarding desirable peer behaviour and penalising the opposite. Desirable behaviours include the uptime of a peer in a mesh and timely forwarding of messages to other peers. Penalised behaviours include the forwarding of invalid messages and the use of the same IP address by many peers. In addition, a hook enables the application to provide feedback on the status of a peer and affect the router's behaviour towards it. The Score Function for each peer is locally computed by the nodes that the peer is connected to and is not shared with other peers (i.e. it is not meant to be a distributed reputation system).

**Opportunistic Grafting:** Peers may get in the same mesh with poorly performing nodes, due to either malicious behaviour or simply poor performance. Mesh replacement has been designed to be slow in order to enable a peer to keep connections with good nodes and avoid a wave of Sybils entering the network and grafting onto nodes. In order to avoid mesh replacement being too slow and add disaster recovery responsiveness to the protocol, we have integrated opportunistic grafting. With opportunistic grafting, each node checks the median score of its mesh peers; if this value dips below an application-controlled threshold, the node sends graft messages to two new peers with scores higher than the current median. Opportunistic grafting takes place every minute and is a recovery measure from a Sybil-dominated network.

### 2.3 Results Summary

After a thorough evaluation in more than 10 different scenarios and many CPU hours spent in a 1920-vCPU cluster, we report that:

**Gossipsub-v1.1 achieves 100% delivery rate**, i.e. it delivers all messages to all nodes in the testing conditions. This is in contrast to Gossipsub-v1.0, which fails to deliver all messages in severe attack cases.

**Gossipsub-v1.1 achieves timely delivery.** In our test network, with 1k honest peers and connection RTTs of 100 ms, we have not found a case where the v1.1 protocol experienced delivery delays higher than 1.6 sec for the 99th percentile of the latency distribution, even in scenarios with Sybil:honest connection ratio as high as 40:1. The maximum latency observed was about 5s but that affected a few messages while the system was recovering from an attack. In contrast, Gossipsub-v1.0 suffers excessive delays (in the order of 10s of seconds) in many cases.

**Flood publishing and adaptive gossip dissemination prove extremely effective mitigation techniques.** Even in cases of large concentration of Sybils (e.g., in attacks against a single target), these techniques manage to bypass Sybil connections and reach honest nodes. At the same time, these techniques do not substantially increase the overall bandwidth, at least when compared to pure flooding. Even in cases of 1k Sybils against a single node, these techniques manage to propagate messages out of the poisoned connection pool of the victim node.

**The Score Function is extremely effective at protecting warm networks from flash attacks.** The score function includes several parameters to reward good behaviour (through high score values) and to penalise misbehaving peers. The combination of parameters of the score function has proven to behave as desired and mitigate a comprehensive series of attacks.

**Opportunistic Grafting is an important disaster recovery mechanism.** Even if Sybils manage to infest the mesh (e.g. in the dawn attack), opportunistic grafting adaptively improves unhealthy meshes in a reasonably fast and responsive manner.

**NOTE:** The Filecoin network block round is 30 sec. In order for all miners to be in sync and mining on the same 'tipset' (i.e., the same group of 5 blocks), all blocks within a tipset have to be delivered to all miners within less than 6 secs from publication. This is a hard deadline for the Filecoin network, as otherwise peers mine on different tipsets, running the risk of a blockchain fork. In our comprehensive test campaign, we have not found any case of the 6-sec deadline being missed by gs-v1.1. In the figures, we visualise the threshold as a vertical blue dashed line (unless the 6-sec deadline falls outside the margin of the plot, in which case we omit the vertical line).

### 3 Recommendations for users

This section includes a series of recommendations for FIL and ETH2 when adopting Gossipsub. These recommendations include considerations for several network sizes, parameter adjustments or recommendations that can be given to different types of nodes.

#### 3.1 FIL

##### Requirements

Stable routing in adversarial conditions: the protocol needs to be able to continue to propagate blocks to the relevant parties in the network (miners) in under 6 sec (ideally closer to 3 sec), no matter what is going on in the network.

The protocol should be robust against the selective routing of Sybils and general high volume spam

*Network numbers (expected):*

- M0 (launch): 1000
- M3: 5000
- M6: 10000
- M12: 30000
- A large portion of miners are likely to be located in China, but it is expected that there will be miners in many other locations too.

*Transaction and Block message size*

- Transactions are from 100 B to 2000 B.
- Block messages are 1 KB plus the size of all CIDs of all messages in that block.

*Expected Message Rate (e.g. per sec or per min)*

- For ‘transaction’ messages, we expect >1 per second
- For blocks, we expect 5 every block time/round, which should be around 30 sec.
- It is worth noting that all of these blocks will come at roughly the same time.

##### Recommendations

We recommend that the filecoin network increases the overlay parameters from the defaults to match our testing parameters:

- $D = 8$
- $D_{score} = D_{low} = 6$
- $D_{high} = 12$
- $D_{lazy} = 12$

This will ensure that even in the face of attack, the stable portion of the mesh will have 6 peers, which is the default value for  $D$ , which is essential for timely delivery as the file network scales upwards to over 30k nodes.

#### 3.2 ETH2.0

##### Requirements

- ETH2 expects no wire protocol changes and do not expect to have to move implementations in lockstep
- Network size 1k to 10k nodes (from ETH1 estimates)
- Phase 0: 70 topics. In subsequent phases, expect  $N * \text{SHARD\_COUNT}$  where SHARD\_COUNT is expected to be 64 and N is a small constant (3 to 8)
- Loads
  - The primary load is 2.5k to 25k attestations (250 B per message) per minute (sent in bursts every 12 seconds, split across 16-64 topics). Expected somewhere between 1/256 to 1/64 nodes on each topic.
  - Additional loads are larger blocks (10 KB to 100 KB). 5 per minute on a single global topic
  - The attestation broadcasting full propagation can likely take in the 3-sec range, followed by a (large) 1-sec buffer with the aggregates being broadcast at 4 sec.

- \* The block 4 sec is split between broadcast and nodes being able to process the block, and potentially create an attestation (sign a message).
- \* Block processing times look sub 1 sec even in extreme cases so block processing *could* take 2 sec, allowing for 1-sec block processing, following by 1-sec buffer to create a message (huge buffer)
- **Summary:** The ETH system ticks every 6 sec, so most transport should complete in 3 sec (with 1-sec buffer) to allow time for validation, crypto, state mutation, and preparation of the next tick.

### Recommendations

**Achieving Block Propagation in under 3 sec:** ETH2 has set a requirement for every block to be propagated in under 3 sec. This is stressing the system, especially given the large number of nodes expected (as the network grows) and the large propagation delays between miners in different continents.

*Recommendation 1:* increasing this to 4-5 sec will guarantee successful delivery according to our extensive testing.

*Recommendation 2:* the Heartbeat setting of Gossipsub is currently at 1 sec. This is a reasonable setting which allows for responsiveness but avoids frantic changes in the protocol behaviour. Smaller values for the heartbeat will result in higher responsiveness (e.g. gossip is emitted at shorter time intervals), but at the expense of higher bandwidth requirements. The ETH2.0 community might want to consider reducing the heartbeat to 600-700 ms, especially with the 3-sec propagation deadline.

*General Recommendation:* This is a general recommendation, not specific to FIL or ETH2.0, but to any stakeholder making use of gs-v1.1 and/or libp2p.

libp2p has been using “secio”,<sup>1</sup> a simple, secure channel protocol,<sup>2</sup> whose main task is to encrypt data transmitted within a session using forward secrecy. The libp2p team is moving away from secio and does not recommend using it. Furthermore, there is a recommendation to move away from RSA, which is what libp2p has been using so far. RSA keys are big and therefore, make any process involving them slow. They induce considerable overhead and therefore, the team is moving towards “ed25519”,<sup>3</sup> which has been shown to be much faster (to generate keys, sign and verify), more lightweight (i.e. induces much less overhead in messages), but does not compromise security.

<sup>1</sup><https://github.com/libp2p/go-libp2p-secio>

<sup>2</sup><https://github.com/libp2p/go-libp2p-secio/issues/7>

<sup>3</sup><https://ed25519.cr.yp.to/>

## 4 Evaluation Setup

### 4.1 Metrics

For each of the test scenarios, we list and present results for selected metrics, which most importantly include:

- **Delivery Rate:** percentage of nodes that receive a message published in the network. Where the delivery rate is not mentioned (all cases for gs-v1.1 and some cases in gs-v1.0), it can be assumed that all messages are successfully delivered.
- **Delivery Delay:** the delay needed for a message to propagate throughout the network and reach all nodes. We provide the CDF and PDF for the delay distributions, and the PDF for messages delayed over p99. We explicitly visualise both the ETH and FIL deadlines at 3 sec and 6 sec, respectively, with a vertical, dashed line. In figures where these thresholds fall outside of the plot, we omit including them.

Where results are worth observing, we are also presenting other metrics related to the internal state of gs-v1.1: score distributions and statistics, both in aggregate and specifically for honest-vs-attacker peers.

A few notes regarding the plots presented in the results sections:

- **Peer Scores Statistics.** In this plot we visualize the min, max, 25th, 75th and 95th percentiles of the score distributions over the entire test run. When either attacker or honest nodes' scores are equal to zero, then this means that those nodes have not been included in the mesh of some observer peer and have not been propagating messages through gossip. Negative score values result out of peers being penalized by the scoring function of some observer.
- **Peer Scores Over Time.** In this plot we visualize the evolution of peer scores on aggregate over the duration of the test run, specifically the median/min in one plot and the min/max in another.
- **CDF and PDF Plots.** The Cumulative Distribution Function (CDF) and Probability Density Function (PDF) plots presented are *not normalized* against the number of messages sent. This was done on purpose in order to be able to see an estimate of the actual number of messages delivered. The normalised plots would have exactly the same shape.
- **PDF 99th Percentile.** We are plotting the Probability Distribution Function (PDF) of the message propagation latency *above* the 99th Percentile (denoted as 'p99'). These plots show the latency distribution for the slowest 1% of messages to propagate through the network (i.e. those messages with the highest 1% of propagation latency).

Most tests last for 3 min and have a warm-up and cool-down period of 30 sec at the beginning and end of the test. The Covert Flash Attack lasts for 5 min in order to allow for both honest and sybil nodes to build score before the attack is carried out. Score function fluctuation during the warm-up and cool-down period should be disregarded as there are no published messages during those periods.

In our test cluster, each container used is assigned 2/3 to 1.2 vCPU and 1 to 2GB of RAM. Honest nodes run as a single peer per container and can thus use the full CPU allotment, while Sybils, which run much simpler code, are packed as multiple peers per container and share this resource.

### 4.2 Default Test Parameters

Unless otherwise noted, the parameters below stay the same throughout our tests.

#### *Network setup*

The publishers are publishing at an aggregate rate of 120 messages per second, with each message sized at 2 KB. The network is configured with 50 ms (10% jitter) latency per edge, making for an RTT of 100 ms for any two nodes.

#### *Gossipsub-v1.1 Mesh Parameters*

```
OVERLAY_D = 8
OVERLAY_DLO = 6
OVERLAY_DHI = 12
```

OVERLAY\_DSCORE = 6  
 OVERLAY\_DLAZY = 12  
 HEARTBEAT = 1s

#### *Honest and Sybil node setup*

N\_NODES [honest] = 1000  
 N\_PUBLISHERS = 100  
 N\_LURKERS = 900  
 N\_DEGREE = 20 [Number of honest connections per honest peer. Note that this is the lower bound of connections for a realistic setting.]

N\_SYBILS = 4000  
 ATTACK\_DEGREE = 100 [Number of connections a Sybil has available to connect to honest peers]

Based on the above, the default Sybil:honest connection ratio is 20.

#### *Score Function Parameters*

##### *Thresholds*

GossipThreshold = -4000.0 [unattainable without invalid message deliveries or app signal]  
 PublishThreshold = -5000.0  
 GraylistThreshold = -10000.0  
 AcceptPXThreshold = 0.0 [PX is not enabled in our test setup – there is no peer routing system]  
 OpportunisticGraftThreshold = 0.0 [this effectively disables by default – it is enabled in certain tests]

#### *Peer Score Parameters*

TopicWeight = 0.25  
 TimeinMeshWeight = 0.0027  
 TimeinMeshQuantum = "1s"  
 TimeinMeshCap = 3600.0  
 FirstMessageDeliveriesWeight = 0.664  
 FirstMessageDeliveriesDecay = 0.9916  
 FirstMessageDeliveriesCap = 1500.0  
 MeshMessageDeliveriesWeight = -0.25  
 MeshMessageDeliveriesDecay = 0.997  
 MeshMessageDeliveriesCap = 400.0  
 MeshMessageDeliveriesThreshold = 10.0  
 MeshMessageDeliveriesActivation = "1m"  
 MeshMessageDeliveryWindow = "5ms"  
 MeshFailurePenaltyWeight = -0.25  
 MeshFailurePenaltyDecay = 0.997  
 InvalidMessageDeliveriesWeight = -99.0  
 InvalidMessageDeliveriesDecay = 0.9994



## 5 Bandwidth Requirements & Cost of Attack

Throughout our test campaign we have used the following settings:

- Message Rate: 120 msg/sec
- Message Size: 2 KB

While messages of 2 KB is a reasonable setting, the message rate we have used is very high for the block topic. This was done on purpose, in order to stress-test the protocol behaviour, but is not a figure we should use to assess the bandwidth requirements when propagating blocks. The figure of 120 msg/sec, however, is not unreasonable to assume for Tx messages. That said, we will assume values closer to the following ones for blocks and transactions, respectively, when calculating bandwidth requirements:

- Block rate: 10 blocks/min (5 blocks per round with round duration 30 sec)
- Transaction rate: 2-4 transactions/second
- Message Size: 2 KB

The formula used to calculate bandwidth requirements is the following:

$$\text{Bandwidth Requirement} = \text{message\_rate} \times \text{message\_size} \times \text{connections}$$

In our tests, we have assumed that honest nodes have 20 connections. In reality, peers might have even more connections but, according to the protocol design, a peer will form a mesh with another 8-12 peers (excluding the peers that it is gossiping with). Assuming 10 blocks/min and for an increasing number of connections, we have:

Number of Connections	Data Vol. (KB/min)	Data Vol. (GB/month)
4	80	3.456
8	160	6.912
12	240	10.368
24	480	20.736
48	960	41.472
96	1920	82.944

Table 1. Blocks Topic: GB/month for Increasing Number of Connections, assuming a rate of 10 new blocks per minute

Number of Connections	Data Vol. (GB/month)
4	41.4
8	82.9
12	124.4
24	248.8
48	497.6
96	995.3

Table 2. Transactions Topic: GB/month for an increasing number of connections (2 Tx/sec)

Number of Tx/sec	Data Vol. (GB/month)
2	124.4
4	248.8
8	497.6
16	995.3
32	1990
64	3981

Table 3. GB/month for an increasing number of Tx (12 connections)

The green cells in the tables are the bandwidth requirement for honest nodes and the red cells are the bandwidth requirement for Sybil nodes. Note that these figures are excluding gossip messages. Furthermore, simple calculations show that the per-second bandwidth requirement for Sybil nodes (*i.e.*, 100 connections each) and for block propagation only (Table 5) is reaching 24 MBs/sec.

- These figures are the upper-bound of bandwidth needed per node. This is because in many attacks (e.g. eclipse, degradation) the sybils act as sinks (that is, they are receiving messages but do not forward further).
- Traditionally, botnets are created by compromised user machines, most commonly connected through home connections. Home connection capacities are usually sufficient for normal DDoS attacks, but a botnet-based attack would be hard to mount in the FIL/ETH2.0 case, as the bandwidth requirement (of 24 MBs/sec or 192 Mbps) for the peers is far higher than the average home-based connection.
- The bandwidth requirement for honest nodes is high but not prohibitive.
- The first two columns of the second table indicate that for 2 Tx/sec and 96 connections per node, a Sybil node requires almost 1 TB of traffic per month.

In terms of cost of attacking the network and according to our AWS-based Testground tests, 1vCPU with 1GB of memory is roughly enough for a Sybil node to open and operate 100 connections. Digital Ocean sells a sufficient plan for \$5/month with a transfer allowance of 1 TB/month. Our rough bandwidth estimates indicate that the bandwidth requirement is higher than 1 TB. The next available option (on Digital Ocean) comes at \$10/month. So the attacks we tried against our 1k node network and ultimately proved unsuccessful would cost in the order of \$40,000/month.

Let us emphasize that in our test campaign **we have not found an attack that can successfully degrade a warm gossipsub v1.1 network**. The only somewhat effective attacks were cold boot attacks, where the attack happens at network formation time, but in all cases the effect was minimal and the network successfully recovered in a short period of time. We are currently focused on testing with ever higher Sybil:honest ratios and trying to find a breaking point, so that we can pinpoint a threshold at which an attacker would be successful. Note however that there is a natural limit in the Sybil:honest ratio that can be attained in practice because of the connection manager component of libp2p which limits the number of connections that a node maintains.

## 6 Establishing the Baseline for Gossipsub

### 6.1 Description

The purpose of this test is to get benchmark values for the operation of Gossipsub under non-adversarial conditions. These benchmark values, especially for some parts of the gs-v1.1 design, will inevitably diverge when under attack. These variations can, therefore, be used as signals of malicious behaviour and trigger mitigation strategies.

### 6.2 Test Description

This is a simple setup of 1000 honest nodes, where we have 100 publishers/miners and 900 lurkers/full nodes. There are no adversarial nodes. The test is intended to set the baseline for gs-v1.1, but also to compare against the vanilla gs-v1.0, that is, without the hardening extensions.

### 6.3 Results

#### Delivery Latency Gossipsub-v1.0 - Baseline Scenario

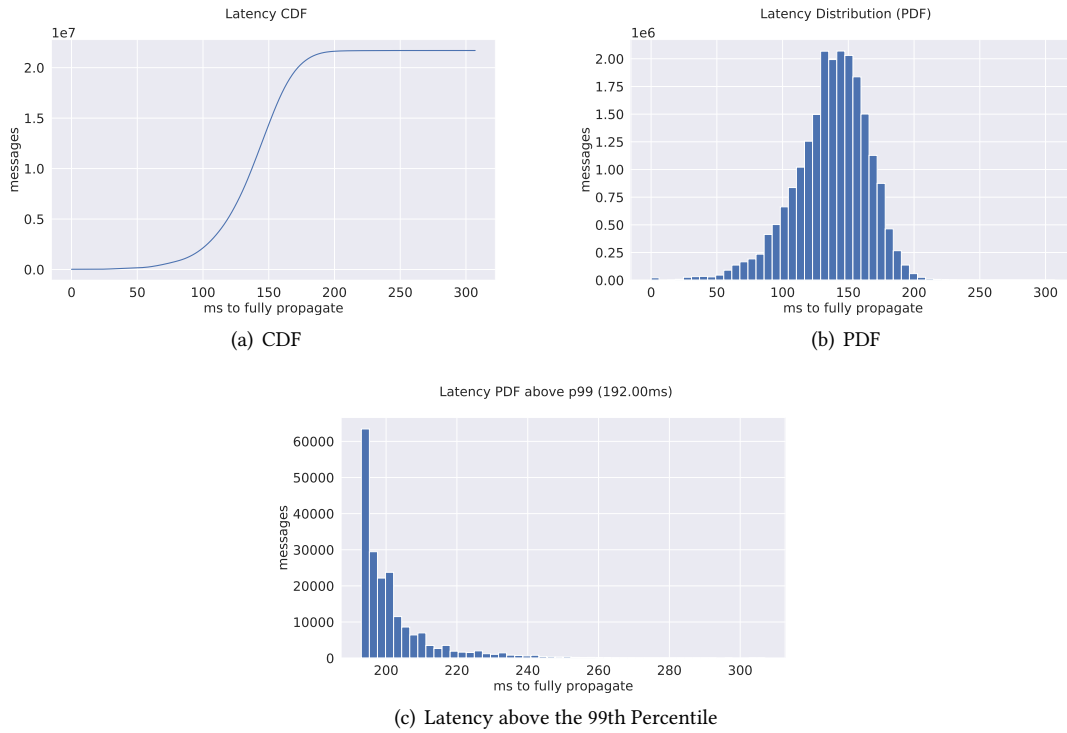


Fig. 1. Gossipsub-v1.0 (a) CDF (b) PDF (c) Latency above the 99th Percentile

### 6.4 Analysis

- For a network of 1000 nodes, the delivery delay of both gs-v1.0 and gs-v1.1 is under 350 ms. gs-v1.0 shows slightly lower overall completion time but this can be attributed to random fluctuation.
- The latency distribution shows that most nodes receive messages under the 150 ms threshold in gs-v1.1 with p99 at 165 ms, while in gs-v1.0 the threshold is 200 ms with p99 at 192 ms. This is because

### Delivery Latency Gossipsub-v1.1 - Baseline Scenario

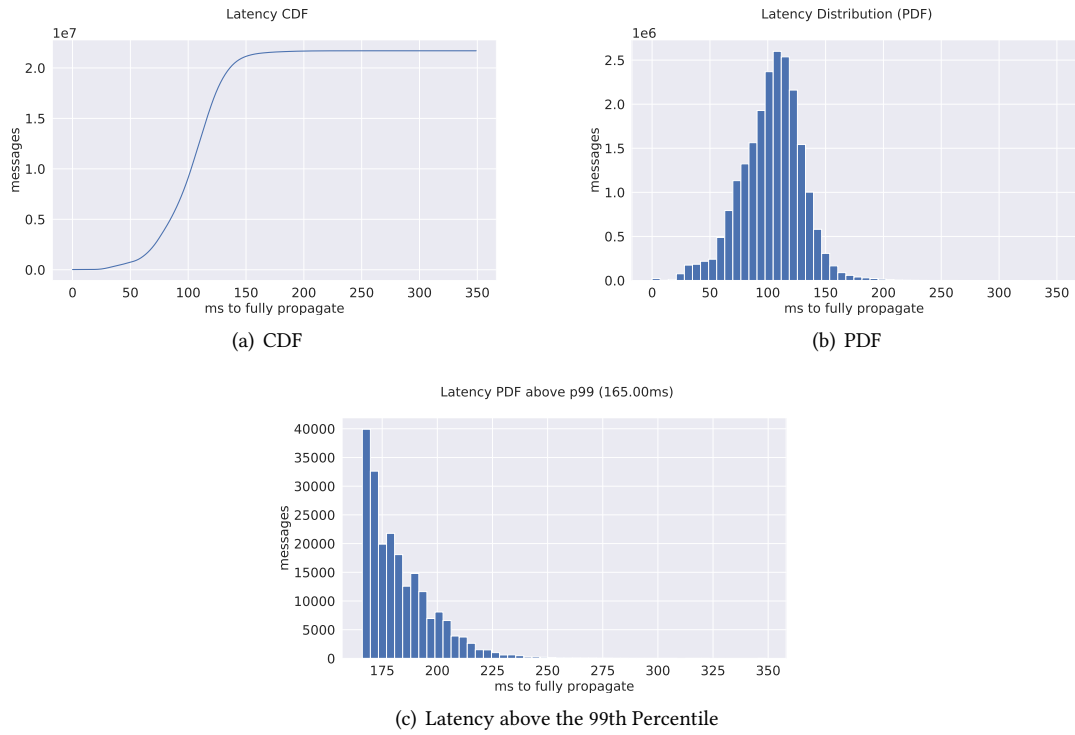


Fig. 2. Gossipsub-v1.1 (a) CDF (b) PDF (c) Latency above the 99th Percentile

gs-v1.1 implements flood publishing, which injects originally published messages to more points in the network.

## 7 Eclipse Attack Against a Single Target

### 7.1 Description of Attack

The general targets of this eclipse attack are:

- to silence a node, i.e. refuse to propagate messages from it
- to distort a node's view of the network, i.e. refuse to propagate messages to it, or propagate wrong or delayed information.

The attack is executed by having a large number of sybils connect to the target and continuously attempt to graft onto its mesh (while respecting the backoff interval).

### 7.2 Mitigation

gs-v1.1's resistance against eclipse attacks is threefold:

- (1) Publishers flood-publishing, which ensures that messages get out through honest node connections.
- (2) Adaptive gossip dissemination, which ensures that messages propagate through gossip with high probability given enough honest node connections.
- (3) Score-influenced peer selection, when pruning due to oversubscription.

In particular, point (3) ensures nodes are accepting just a few new/low scoring nodes in the mesh, keeping a portion of their connections grafted to their old/high scoring peers. This ensures that an army of sybils attacking a warm node will only succeed in occupying a few slots in the mesh. In turn, this means that the node is not silenced and can propagate messages to the healthy connections of the mesh quickly.

### 7.3 Test Description

In this test case, we have 1k honest nodes (100 publishers and 900 lurkers) as in the baseline test. In addition, we have 1k Sybils targeting a single publisher. The attack is attempted against a warm network: the attack nodes are introduced 1 min into the test, while the warm-up period is 30 sec.

### 7.4 Results

#### 7.4.1 Delivery Rate.

	Gossipsub-v1.0	Gossipsub-v1.1
Peers	1000	1000
Published Messages	21700	21700
Delivered Messages	21633067	21700000
Delivery Rate	99.6%	100%

#### 7.4.2 Delivery Latency. See Figs. 9 and 10

### 7.5 Analysis

The most salient feature of this attack against gs-v1.0 is that it is successful in silencing the target node. Once the attack starts, a significant number of its messages are not delivered, while the delivery delay for those that make it through is as high as 4 sec.

In contrast, gs-v1.1 is largely immune. All messages are delivered, and while the delivery delay for a few messages does increase to up to 600 ms, this can be attributed to the increased CPU usage of the targeted publisher, as it now has to flood-publish to 1k+ peers at over 1 MB/sec.

The effects of the attack are largely visible in the p99 latency plots, as a single publisher accounts for exactly 1% of the published messages in our test setup.

### Delivery Latency Gossipsub-v1.0 - Eclipse Single Target

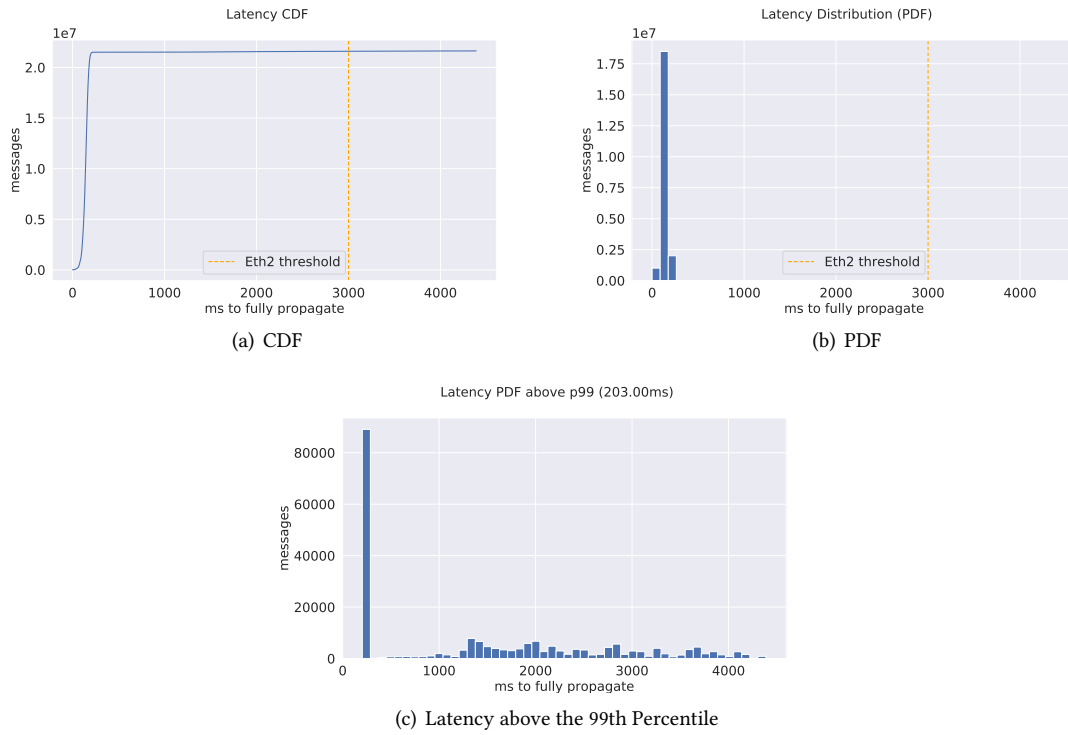


Fig. 3. Gossipsub-v1.0 (a) CDF (b) PDF (c) Latency above the 99th Percentile

### Delivery Latency Gossipsub-v1.1 - Eclipse Single Target

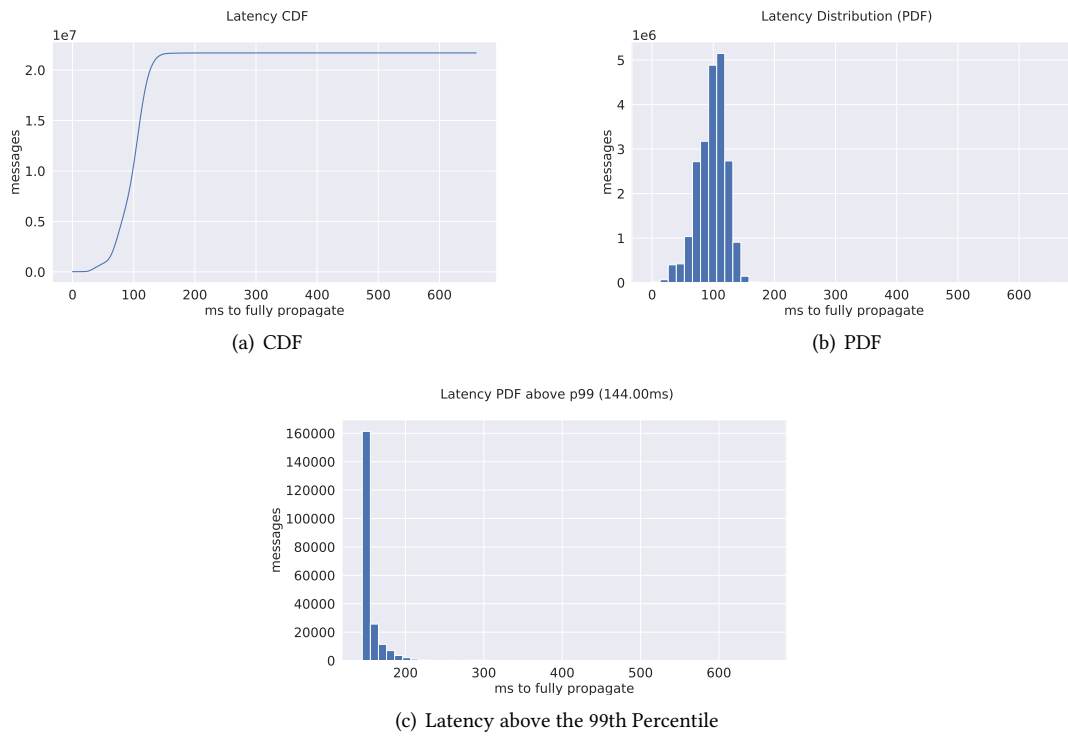


Fig. 4. Gossipsub-v1.1 Delivery Latency (a) CDF (b) PDF (c) Latency above the 99th Percentile

## 8 Eclipse Attack Against the Entire Network

### 8.1 Description of Attack

This is an extension of the eclipse attack against a single node, whereby the attacker generates a large number of Sybils and targets the entire network. The Sybils attack every honest peer in the network in an attempt to disconnect it and completely disrupt message propagation. The attack is carried out by the Sybils connecting to every honest peer and continuously grafting onto the mesh.

### 8.2 Mitigation

The three-fold defensive mechanisms of gs-v1.1 apply in the same way as the single-target eclipse attack: flood publishing, adaptive gossip dissemination, and peer scoring all combine to thwart the attack against a warm network.

### 8.3 Test Description

In this test, we have 1k honest nodes (100 publishers and 900 lurkers) as in the baseline test, but in addition we have 4k Sybils targeting every single node in the network by establishing 100 honest connections each. The Sybil:honest connection ratio is 20:1 for the average node.

The attack is attempted against a warm network; the attack nodes are introduced 1 min into the test, while the warm-up period is 30 sec.

### 8.4 Results

#### 8.4.1 Delivery Rate.

	Gossipsub-v1.0	Gossipsub-v1.1
Peers	1000	1000
Published Messages	21700	21700
Delivered Messages	21698016	21700000
Delivery Rate	99.9%	100%

#### 8.4.2 Delivery Latency. See Figs. 5 and 6

#### 8.4.3 Peer Score Values over Time. See Fig. 7

#### 8.4.4 Peer Scores. See Fig. 8

### 8.5 Analysis

The results show that gs-v1.0 is severely affected by the attack. Once the attack starts, the system starts losing messages and those that are delivered have very high latency, with a distribution that approximates a normal distribution centred at 8 sec. This indicates that the attack was successful at taking over the mesh and all successful message propagation is happening almost purely through gossip.

In contrast, gs-v1.1 is largely immune to the attack. The attackers only manage to occupy a few slots in the mesh, which doesn't make any dent to the delivery rate and only slightly affects the overall latency with p99 at 181 ms; the maximum delay is 1.2 sec.

Most attacker nodes do not get into the mesh (see zero score value in Fig. 8) and those that manage to get into the mesh get negative values and are subsequently pruned from the mesh.



### Delivery Latency Gossipsub-v1.0 - Eclipse Entire Network

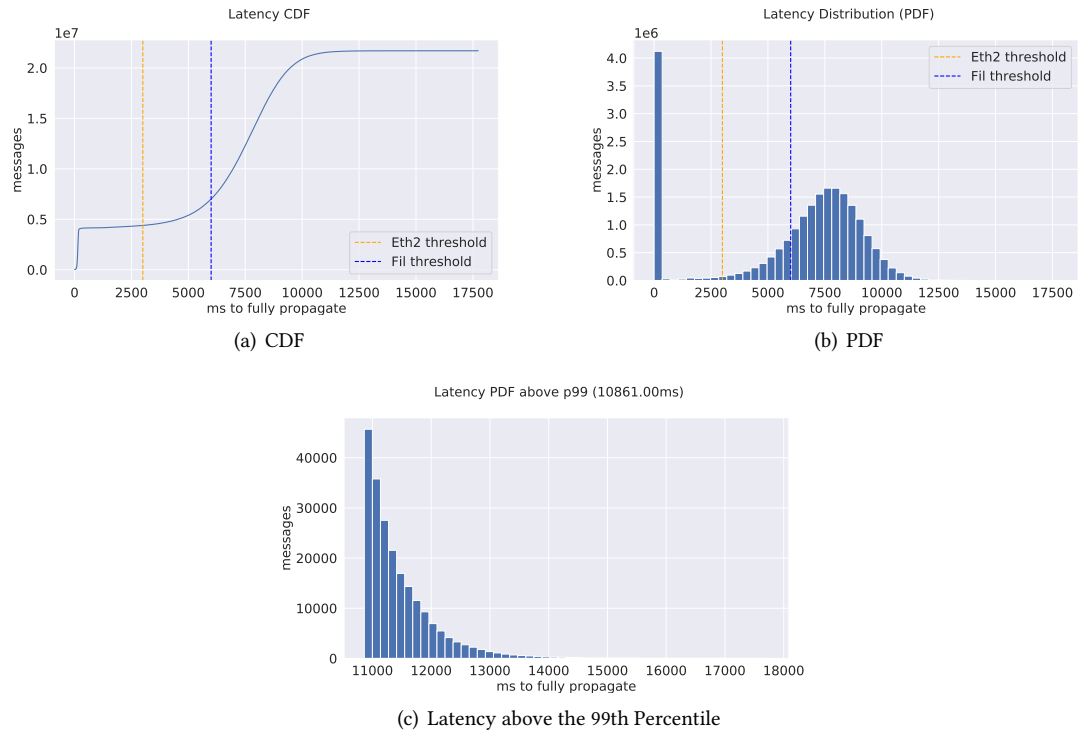


Fig. 5. Gossipsub-v1.0 (a) CDF (b) PDF (c) Latency above the 99th Percentile

### Delivery Latency Gossipsub-v1.1 - Eclipse Entire Network

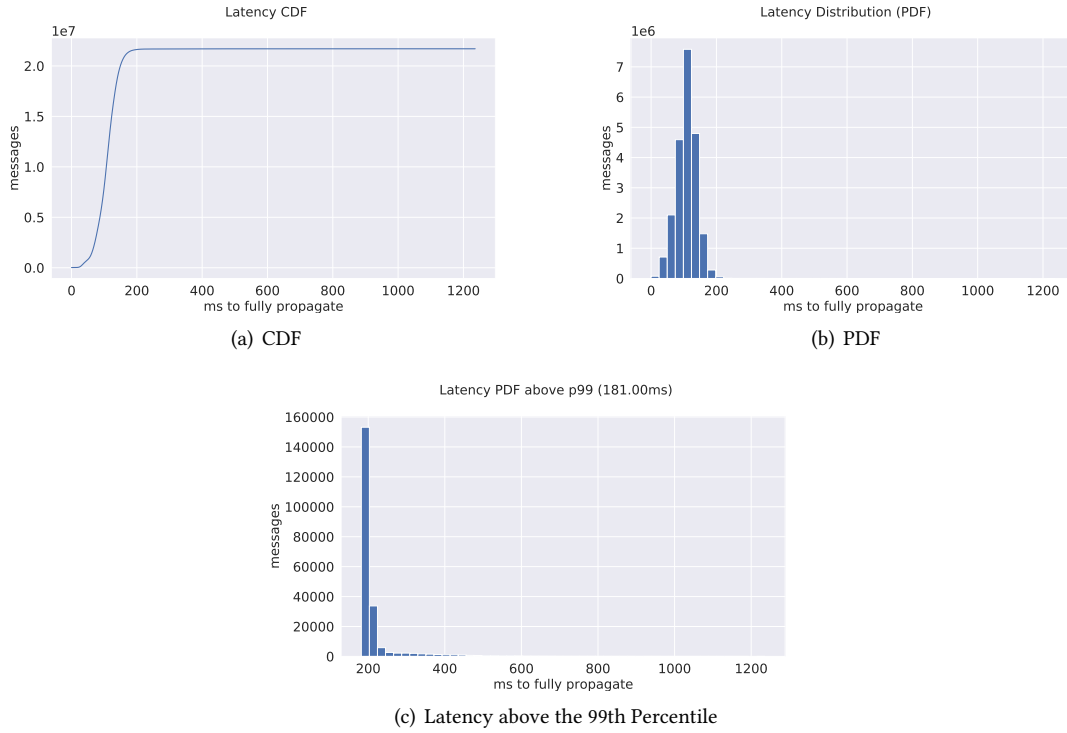


Fig. 6. Gossipsub-v1.1 Delivery Latency (a) CDF (b) PDF (c) Latency above the 99th Percentile

### Peer Score Values Over Time - Eclipse Entire Network

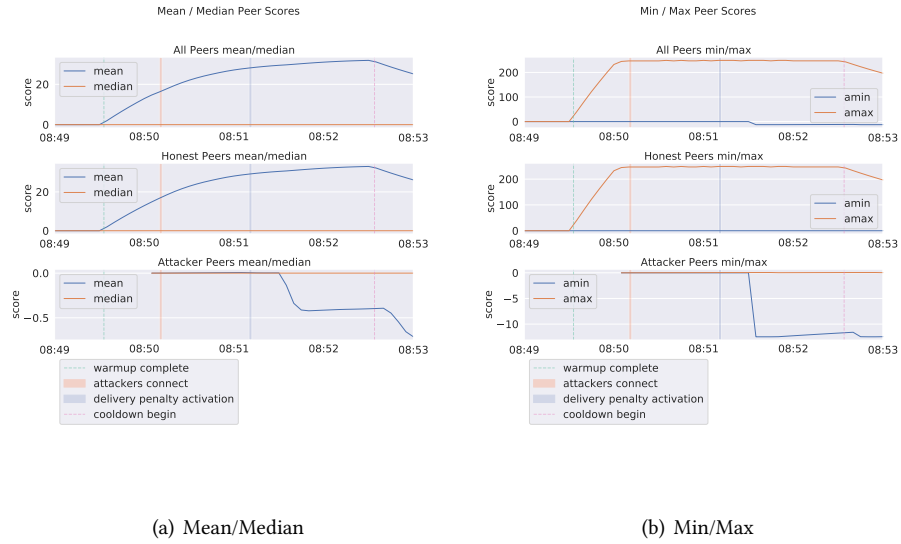


Fig. 7. Gossipsub-v1.1 Peer Score Values over Time (a) Mean/Median (b) Min/Max

### Peer Scores - Eclipse Entire Network

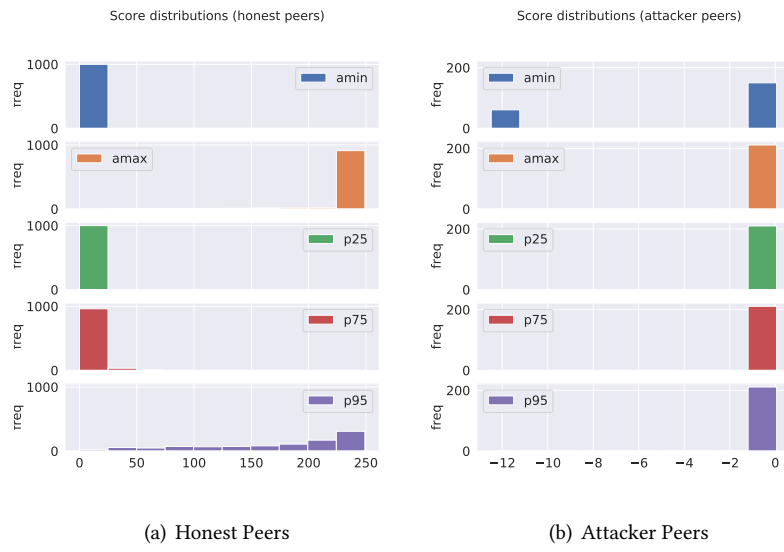


Fig. 8. Gossipsub-v1.1 Peer Score Distributions (a) Honest Peers (b) Attacker Peers

## 9 Eclipse Attack Against all Publishers

### 9.1 Description of Attack

This is another variation of the eclipse attack, where Sybils are only targeting the publisher nodes. The attack is designed in order to investigate what is the effect of a large number of attackers targeting all miners in a blockchain application.

### 9.2 Mitigation

Flood-publishing and increased gossip propagation are again techniques that help publisher nodes propagate messages and overcome Sybil-dominated mesh connections. Peer scoring, and specifically the  $P_3/P_{3b}$  mesh delivery penalty also come into play as they help the targeted nodes eventually evict Sybils from their view of the mesh for lack of contribution.

### 9.3 Test Description

Similarly to the previous tests, here, we have 1k honest nodes (100 publishers and 900 lurkers), where each of the honest nodes can establish 20 connections. In addition, there are 4k Sybils establishing 100 connections each. However, in contrast to the previous tests, here, all Sybil connections are targeting the 100 publishers only. This increases the Sybil:honest connection ratio is 40:1 for the average node, as compared to 20:1 in the majority of the previous tests.

The attack is attempted against a warm network; the attack nodes are introduced 1 min into the test, while the warm-up period is 30 sec.

### 9.4 Results

#### 9.4.1 Delivery Rate.

	Gossipsub-v1.0	Gossipsub-v1.1
Peers	1000	1000
Published Messages	21700	21700
Delivered Messages	15406525	21700000
Delivery Rate	70.9%	100%

#### 9.4.2 Delivery Latency. See Figs. 9 and 10

#### 9.4.3 Peer Score Values over Time. See Fig. 11

#### 9.4.4 Peer Scores. See Fig. 12

### 9.5 Analysis

The attack is devastating against gs-v1.0, which delivers only 70% of the messages and those that get through do so with prohibitive latency.

In contrast, gsv1.1 is immune to the attack, for two reasons:

- Flood publishing ensures that the publishers get their messages out to the wider network by virtue of honest connections.
- The core of the mesh, where lurkers connect to each other in order to propagate messages in the wider network is unaffected by the attack, as Sybils are not targeting them.
- The fact that lurker nodes are not targeted by Sybils also contributes towards fast propagation of messages, as most nodes in the network (remember lurker nodes are 900 out of 1,000) do not need to get into gossip propagation rounds (which by default are slower).

The end result is that the attack is completely ineffective in silencing the publishers.

Note that this test was run with 1.2 vCPU allotment, in contrast to the single publisher eclipse attack which had 2/3 vCPU allotment. This explains why there was no spike in latency (the single publisher attack

### Delivery Latency Gossipsub-v1.0 - Eclipse Publisher Nodes

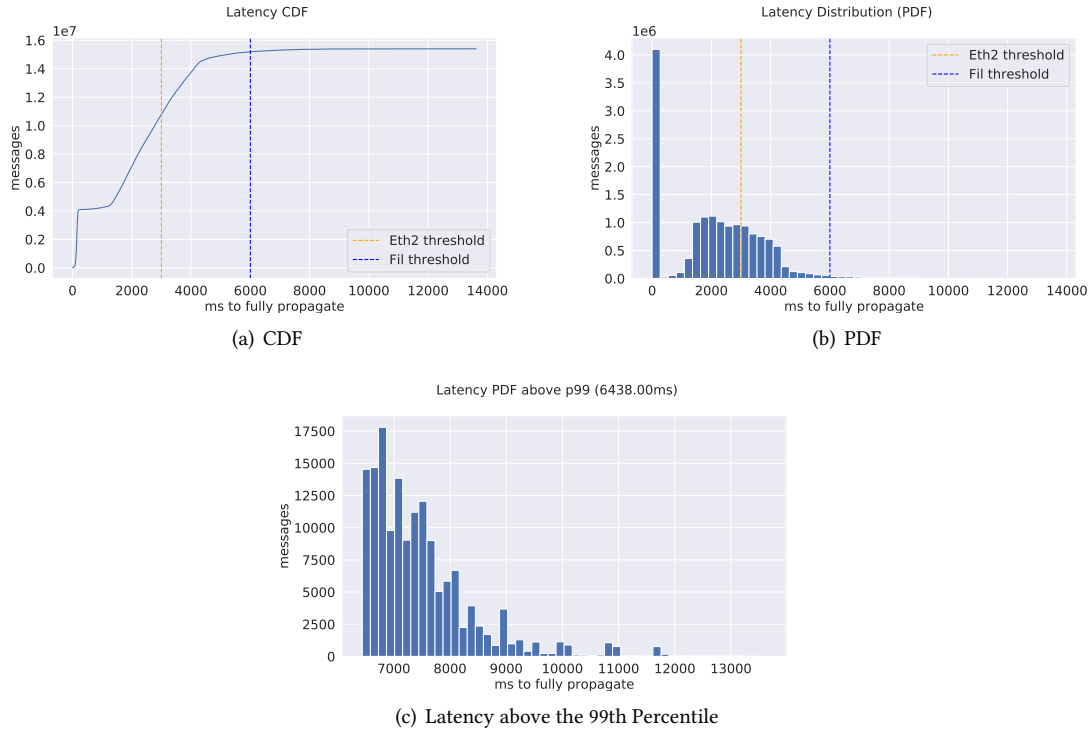


Fig. 9. Gossipsub-v1.0 (a) CDF (b) PDF (c) Latency above the 99th Percentile

maximum delay spiked at a little over 600ms), consistent with our interpretation of the previous results. This result prompted us to increase the vCPU allotment for remaining attacks: tests up to this point were run with 2/3 vCPU and 1GB of RAM, while this and all subsequent attacks were run with 1.2 vCPUs and 2GB of RAM.

### Delivery Latency Gossipsub-v1.1 - Eclipse Publisher Nodes

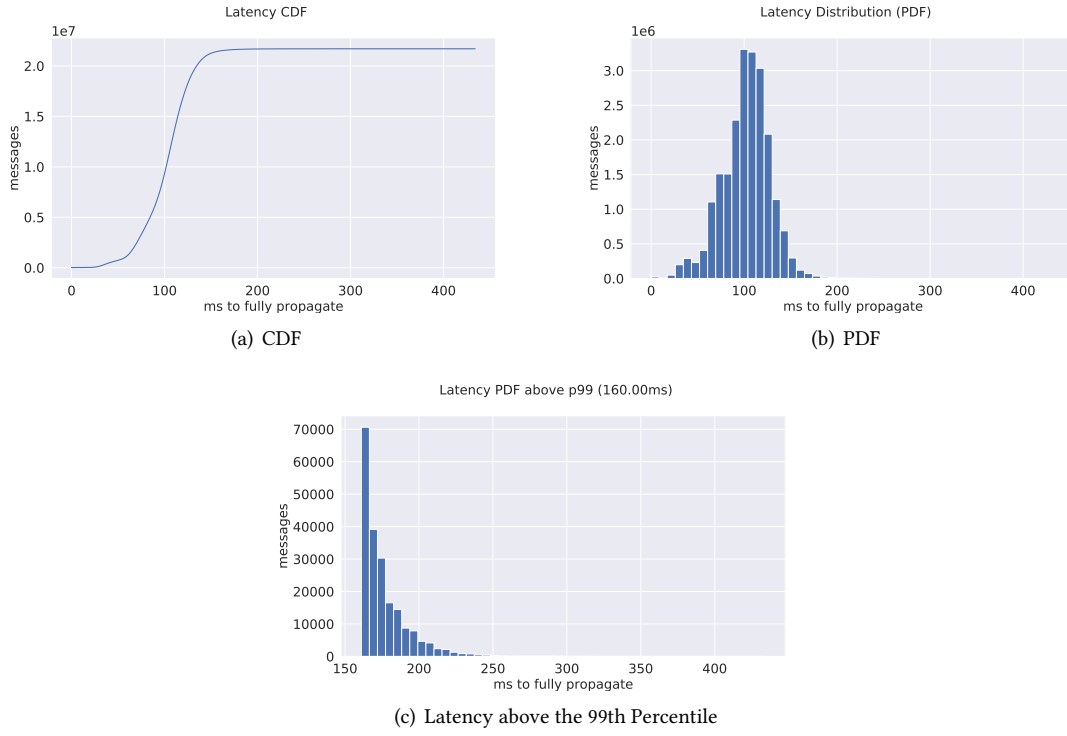


Fig. 10. Gossipsub-v1.1 Delivery Latency (a) CDF (b) PDF (c) Latency above the 99th Percentile

### Peer Score Values Over Time - Eclipse Publisher Nodes

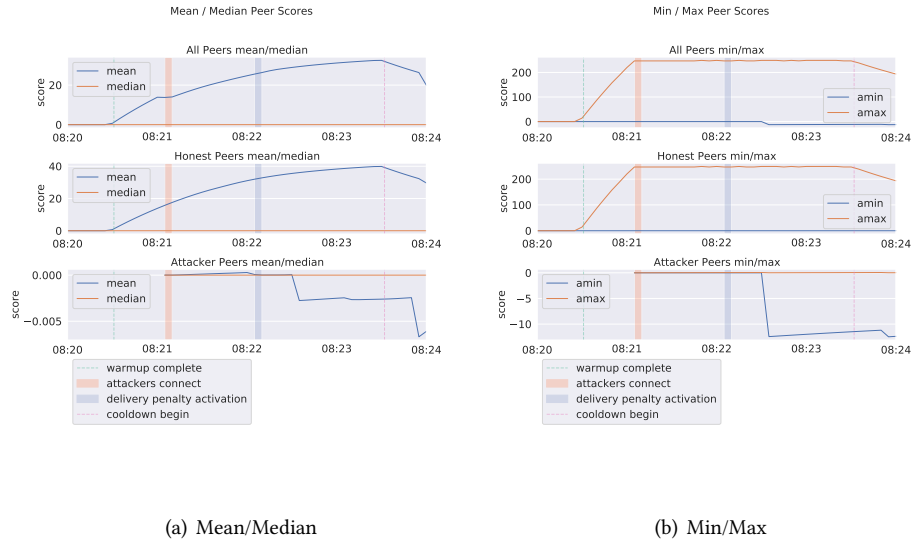


Fig. 11. Gossipsub-v1.1 Peer Score Values over Time (a) Mean/Median (b) Min/Max

### Peer Scores - Eclipse Publisher Nodes

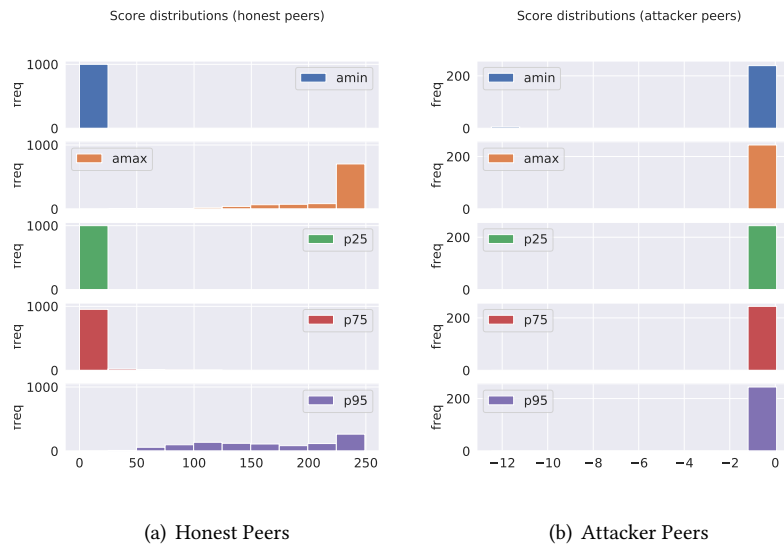


Fig. 12. Gossipsub-v1.1 Peer Score Distributions (a) Honest Peers (b) Attacker Peers

## 10 Censor Attack Against a Single Target

### 10.1 Description of Attack

In this attack, Sybils seek to establish themselves in the mesh and propagate all messages except those published by the target peer. The objective of the attacker is to censor the target and prevent its messages from reaching the rest of the network.

The attack is carried out by the sybils connecting to all honest peers in the network and continuously attempting to graft onto the mesh. Once grafted, they forward all messages except for messages originating from the target.

A salient feature of this attack is that the sybils appear to behave properly from almost all vantage points, but hairpin drop the message of the target. This attack is difficult to detect by scoring, as the sybils build up score by virtue of propagating all other messages.

### 10.2 Mitigation

The main mitigations against a successful censor attack are flood publishing and adaptive gossip dissemination. Even if the attackers do succeed in taking over the entire mesh, the system can still propagate the target's messages as long as there is a sufficient number of connections between honest peers. In addition, a warm network can defend itself against the attack by virtue of  $D\_score$  selection: the attackers should fail to occupy more than a few slots even though they build score by virtue of message propagation.

### 10.3 Test Description

In this test, we have 1k honest nodes (100 publishers and 900 lurkers) as in the baseline test. In addition, we have 2k Sybils connecting to every single node in the network by establishing 100 honest connections each. The Sybil:honest connection ratio is 10:1 for the average node.

The attack is attempted against a warm network: the attack nodes are introduced 1 min into the test, while the warm-up period is 30 sec.

### 10.4 Results

#### 10.4.1 Delivery Rate.

	Gossipsub-v1.0	Gossipsub-v1.1
Peers	1000	1000
Published Messages	21700	21700
Delivered Messages	21699989	21700000
Delivery Rate	99.9%	100%

#### 10.4.2 Delivery Latency. See Figs. 13 and 14

#### 10.4.3 Peer Score Values over Time. See Fig. 15

#### 10.4.4 Peer Scores. See Fig. 16

### 10.5 Analysis

The censorship attack is successful against Gossipsub-v1.0, as the attackers take over the mesh and establish themselves as the source of truth. Messages from the target propagate almost purely through gossip, which results in some message loss and high latency.

On the other hand, Gossipsub-v1.1 successfully defends against the attack by virtue of restricting the attackers to a few slots in the mesh. Even with their better connectivity and score build-up, they only succeed in establishing themselves in the mesh of a few peers beyond the slots allotted by  $D\_score$ . The end result is that propagation of the target's messages is mostly unaffected, even though there is a longer tail with latency reaching up to 1.1 sec, which is indicative of some gossip propagation (note that gossip propagation takes



### Delivery Latency Gossipsub-v1.0 - Censor Single Target

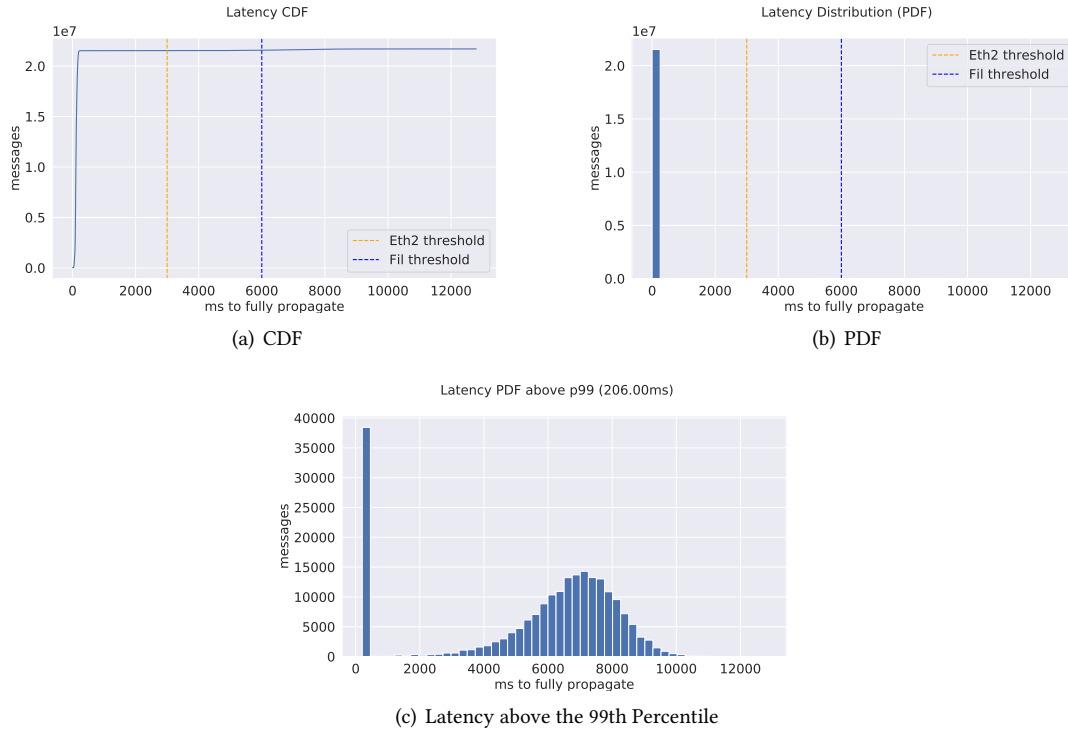


Fig. 13. Gossipsub-v1.0 (a) CDF (b) PDF (c) Latency above the 99th Percentile

place in rounds after every heartbeat and is much slower than propagating messages with peers in the mesh); this can be attributed to the occasional dropped RPC that has been observed in our test network. At the same time, the propagation latency of other peers' messages decreases, which improves the performance of the network as a whole.

### Delivery Latency Gossipsub-v1.1 - Censor Single Target

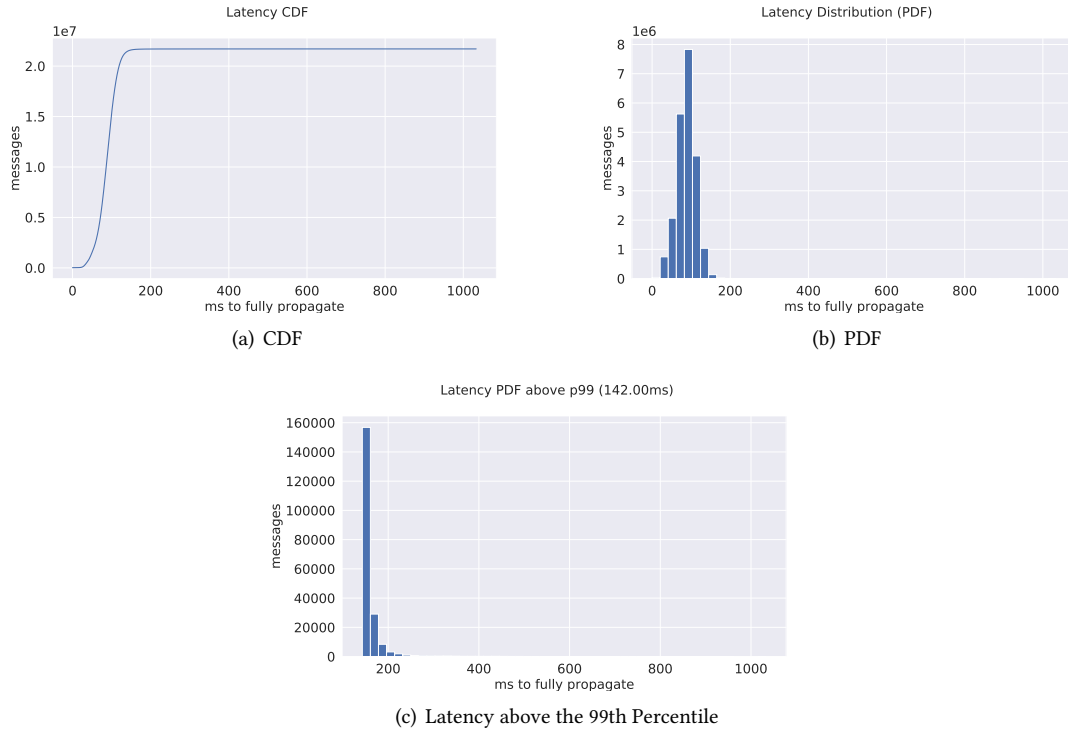


Fig. 14. Gossipsub-v1.1 Delivery Latency (a) CDF (b) PDF (c) Latency above the 99th Percentile

### Peer Score Values Over Time - Censor Single Target

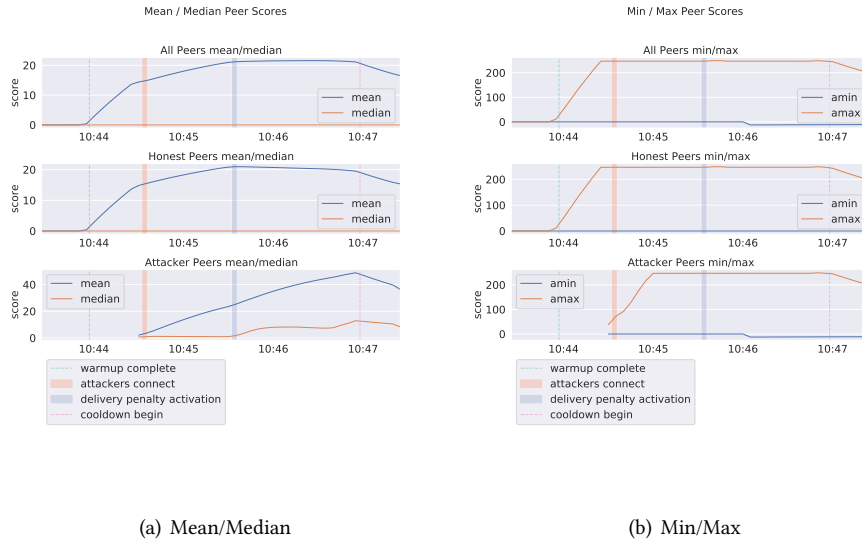


Fig. 15. Gossipsub-v1.1 Peer Score Values over Time (a) Mean/Median (b) Min/Max

### Peer Scores - Censor Single Target

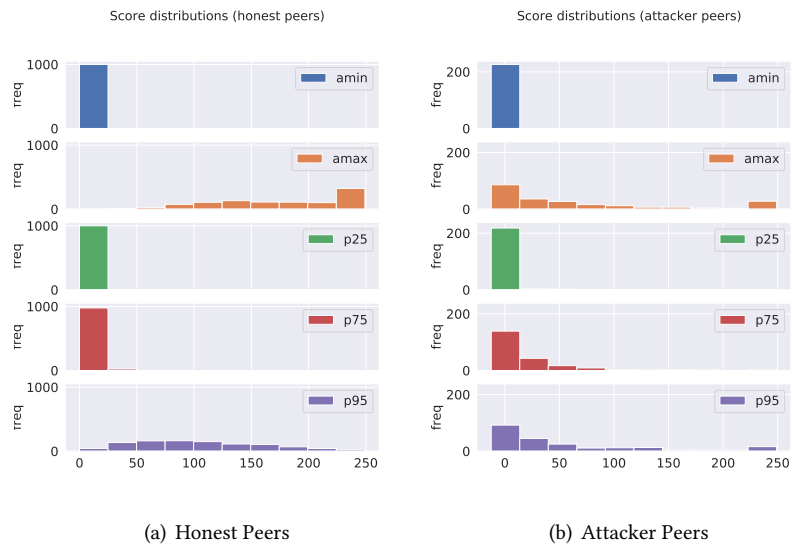


Fig. 16. Gossipsub-v1.1 Peer Score Distributions (a) Honest Peers (b) Attacker Peers

## 11 Network Degradation Attack

### 11.1 Description of Attack

In this attack Sybils seek to establish themselves in the mesh, but intentionally drop messages with some probability. The objective of the attacker is to degrade the network and introduce artificial delays or delivery failure.

The attack is carried out by the Sybils connecting to all honest peers in the network and continuously attempting to graft onto the mesh. Once grafted, they drop messages with Degradation Probability while still forwarding the remaining messages.

### 11.2 Mitigation

Flood publishing and increased gossiping are the main mitigation strategies here. Furthermore, for high degradation factors, the  $P_3, P_{3b}$  parameters of the scoring function can kick in and boot the Sybils from the mesh. In addition, a warm network can defend itself by virtue of the  $D_{score}$  parameters, whereby Sybils can only occupy a few slots in the mesh.

### 11.3 Test Description

In this Test Case, we have 1k honest nodes (100 publishers and 900 lurkers) as in the baseline test, but in addition we have 2k Sybils targeting every single node in the network by establishing 100 honest connections each. The Sybil:honest connection ratio is 10:1 for the average node.

The attack is attempted against a warm network: the attack nodes are introduced 1min into the test, while the warm-up period is 30s.

### 11.4 Results

#### 11.4.1 Delivery Rate

##### Degradation Probability 0.1

	Gossipsub-v1.0	Gossipsub-v1.1
Peers	1000	1000
Published Messages	21700	21700
Delivered Messages	21700000	21700000
Delivery Rate	100%	100%

##### Degradation Probability 0.5

	Gossipsub-v1.0	Gossipsub-v1.1
Peers	1000	1000
Published Messages	21700	21700
Delivered Messages	21699997	21700000
Delivery Rate	99.9%	100%

##### Degradation Probability 0.9

	Gossipsub-v1.0	Gossipsub-v1.1
Peers	1000	1000
Published Messages	21700	21700
Delivered Messages	21699870	21700000
Delivery Rate	99.9%	100%

11.4.2 *Delivery Latency.* See Figs. 17, 19 and 21 for gs-v1.0, 18, 20, and 22 for gs-v1.1

11.4.3 *Peer Score Values over Time.* See Fig. 23, 24 and 25

11.4.4 *Peer Scores.* See Fig. 26, 27, and 28

### Delivery Latency Gossipsub-v1.0 - Degrade Probability 0.1

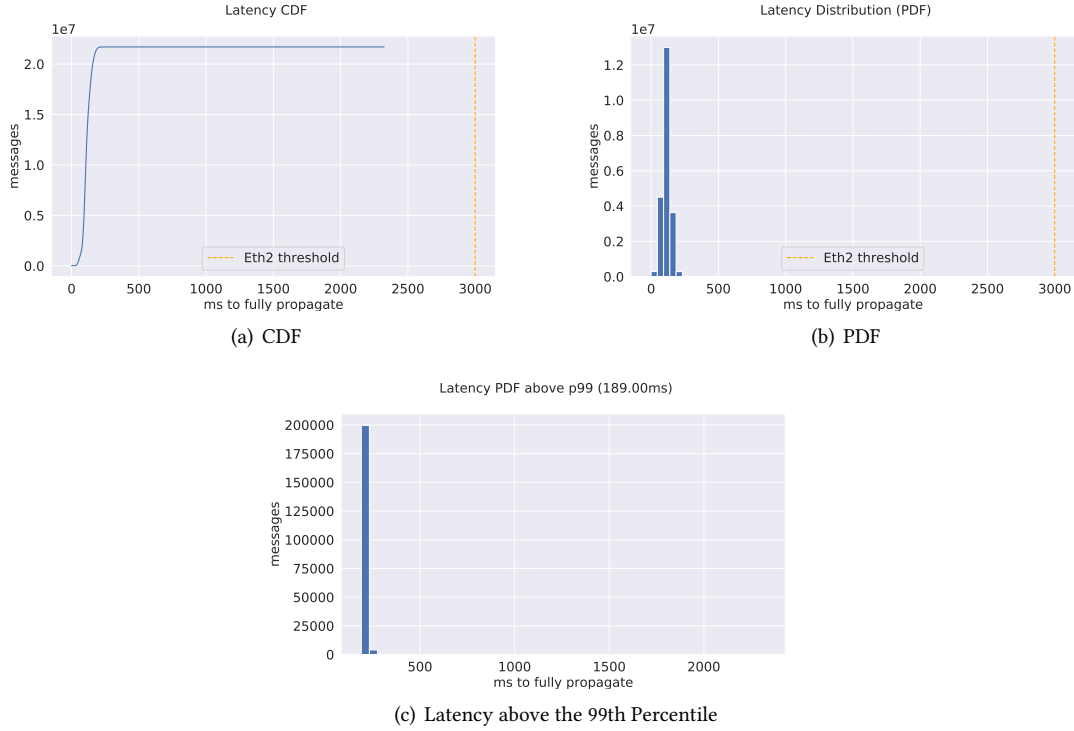


Fig. 17. Gossipsub-v1.0 (a) CDF (b) PDF (c) Latency above the 99th Percentile

## 11.5 Analysis

With degradation probability of 0.1, the attack is ineffective even against gs-v1.0. The reason is simple: the message drops are uncoordinated and happen with low enough probability that the Sybils complement each other. There was some gossip propagation for a few messages in gs-v1.0 that suggests that the attack had a small effect, but not enough to make a dent. gs-v1.1 is completely immune to it; interestingly, the sybils end up augmenting the network thanks to their high connectivity and improve overall latency.

The attack starts to become effective against gs-v1.0 with degradation probability of 0.5 and becomes very effective with degradation probability of 0.9, as the system fails to deliver all messages and latency increases to prohibitive levels.

gs-v1.1, on the other hand, remains immune to the attack even with 0.9 degradation probability. This is because the Sybils don't manage to occupy enough slots in the mesh of honest nodes, even though they achieve positive scores by virtue of message propagation. The worst effect that was observed was that a few messages got delayed by up to 1 sec, which is indicative of delivery through gossip propagation. Recall that gossip is emitted upon every heartbeat, which is set to 1 sec in our tests.

### Delivery Latency Gossipsub-v1.1 - Degrade Probability 0.1

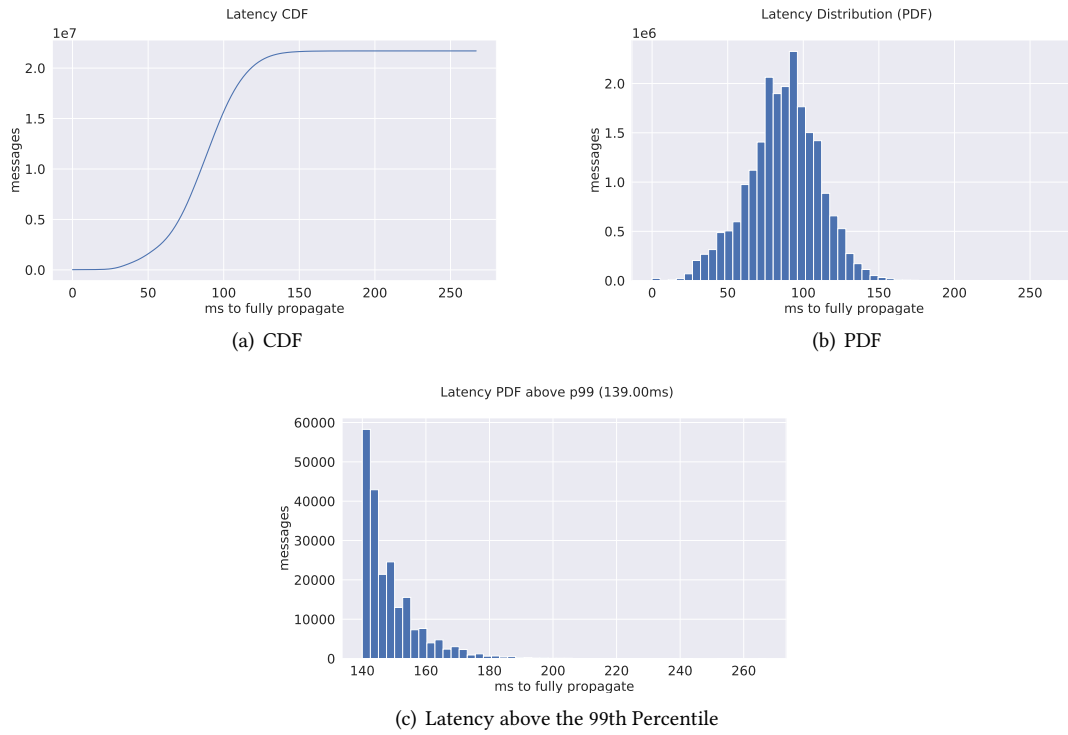


Fig. 18. Gossipsub-v1.1 Delivery Latency (a) CDF (b) PDF (c) Latency above the 99th Percentile

### Delivery Latency Gossipsub-v1.0 - Degrade Probability 0.5

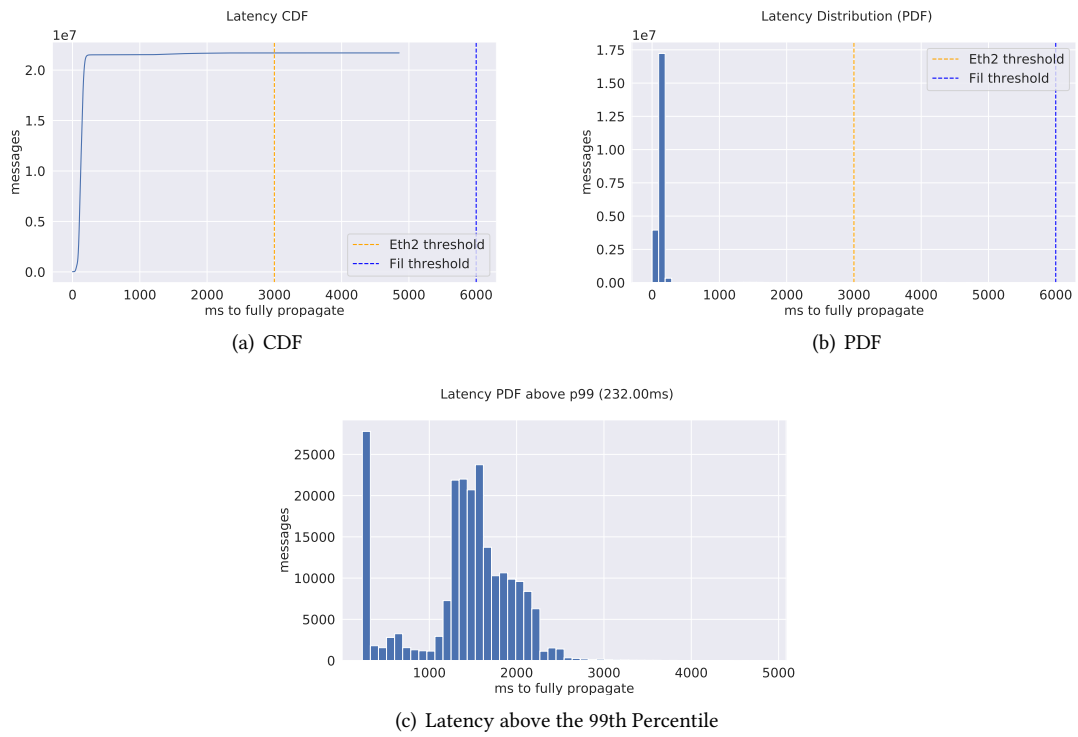


Fig. 19. Gossipsub-v1.0 (a) CDF (b) PDF (c) Latency above the 99th Percentile

### Delivery Latency Gossipsub-v1.1 - Degrade Probability 0.5

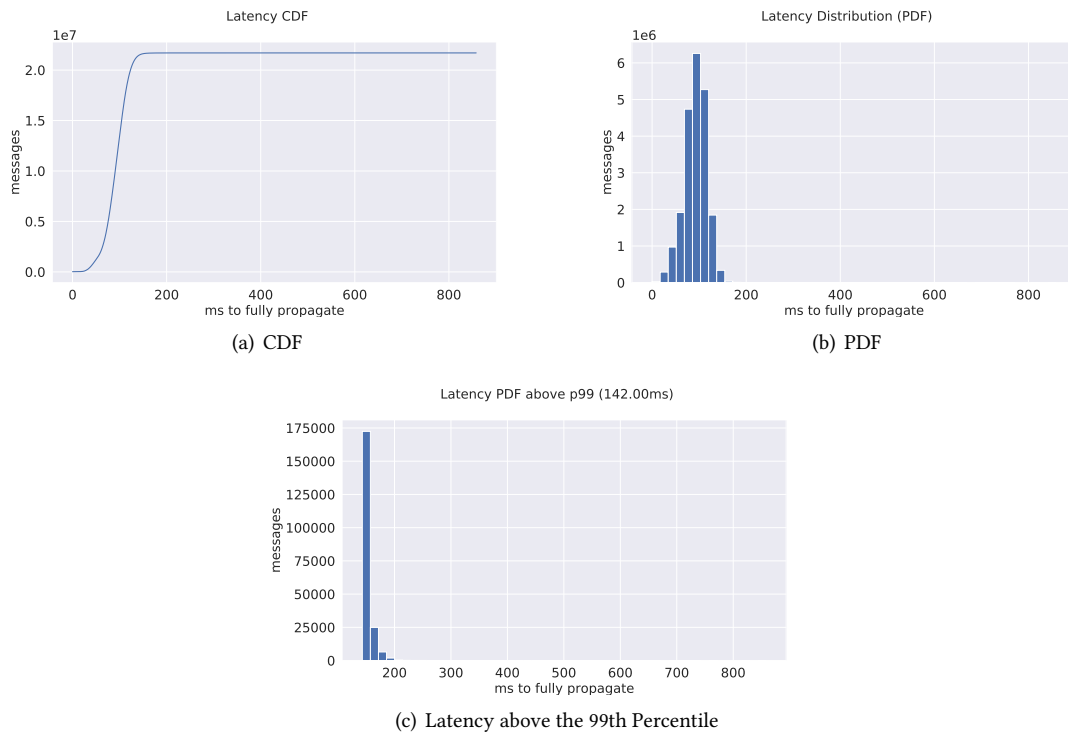


Fig. 20. Gossipsub-v1.1 Delivery Latency (a) CDF (b) PDF (c) Latency above the 99th Percentile



### Delivery Latency Gossipsub-v1.0 - Degrade Probability 0.9

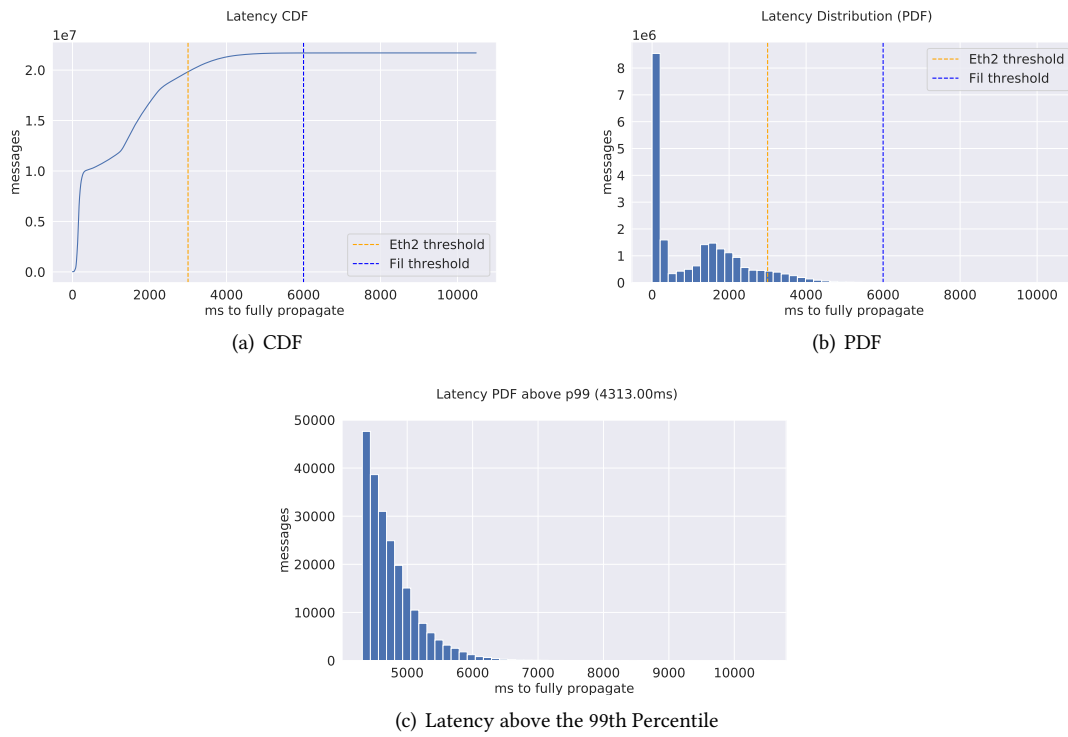


Fig. 21. Gossipsub-v1.0 (a) CDF (b) PDF (c) Latency above the 99th Percentile

### Delivery Latency Gossipsub-v1.1 - Degrade Probability 0.9

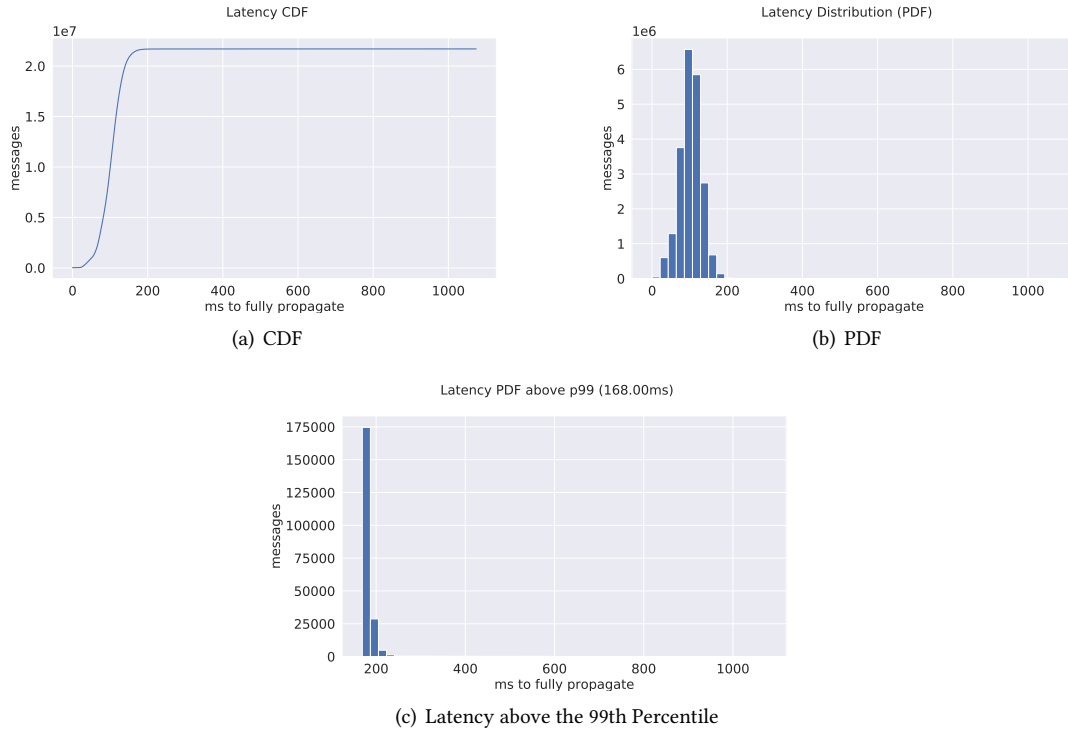


Fig. 22. Gossipsub-v1.1 Delivery Latency (a) CDF (b) PDF (c) Latency above the 99th Percentile

### Peer Score Values Over Time - Degrade Probability 0.1

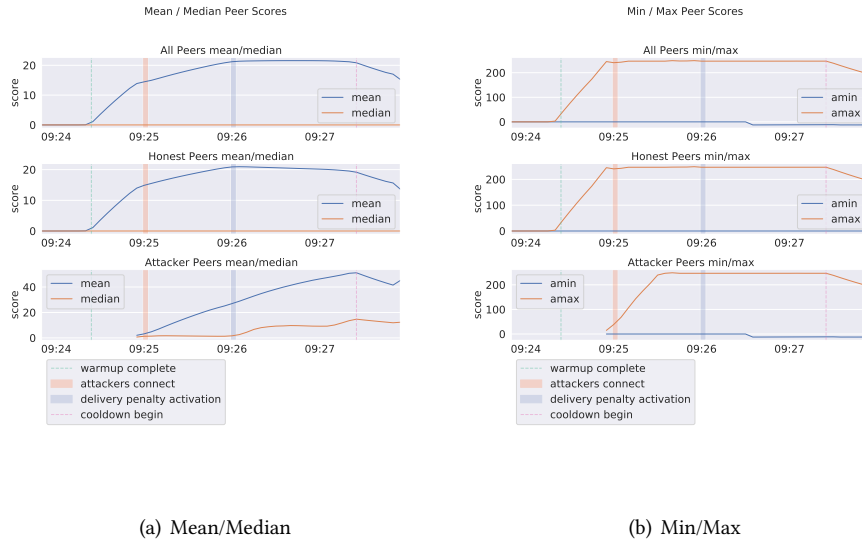


Fig. 23. Gossipsub-v1.1 Peer Score Values over Time (a) Mean/Median (b) Min/Max

### Peer Score Values Over Time - Degrade Probability 0.5

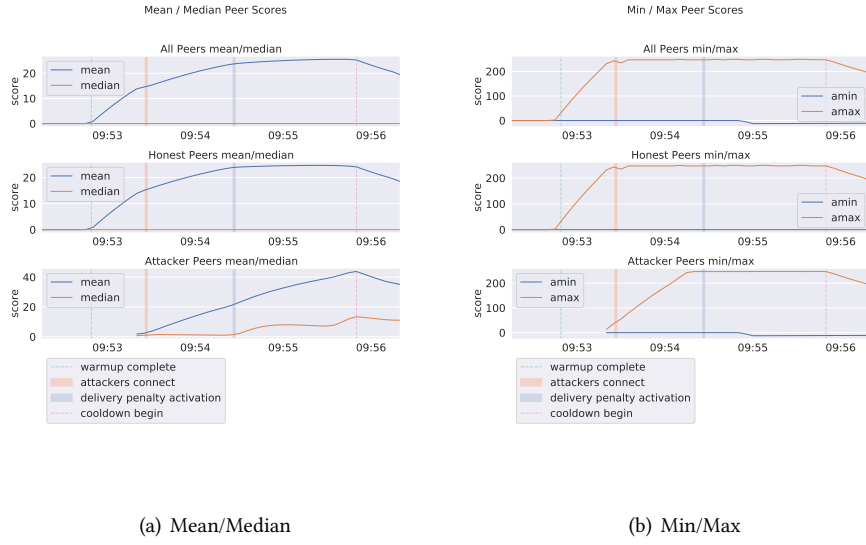


Fig. 24. Gossipsub-v1.1 Peer Score Values over Time (a) Mean/Median (b) Min/Max

### Peer Score Values Over Time - Degrade Probability 0.9

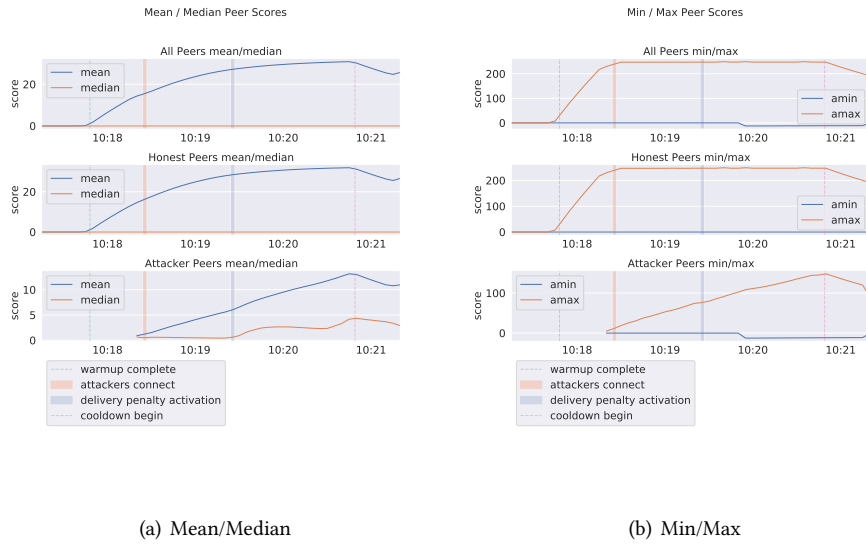


Fig. 25. Gossipsub-v1.1 Peer Score Values over Time (a) Mean/Median (b) Min/Max

### Peer Scores - Degrade Probability 0.1

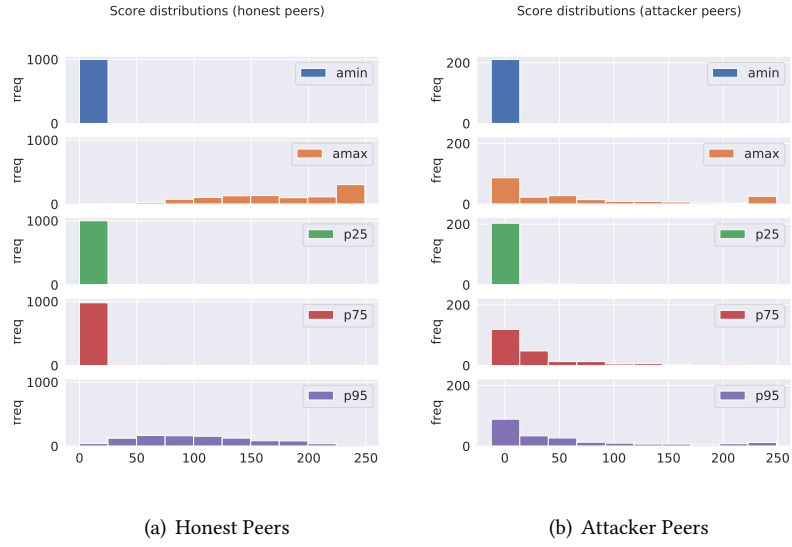


Fig. 26. Gossipsub-v1.1 Peer Score Distributions (a) Honest Peers (b) Attacker Peers

### Peer Scores - Degrade Probability 0.5

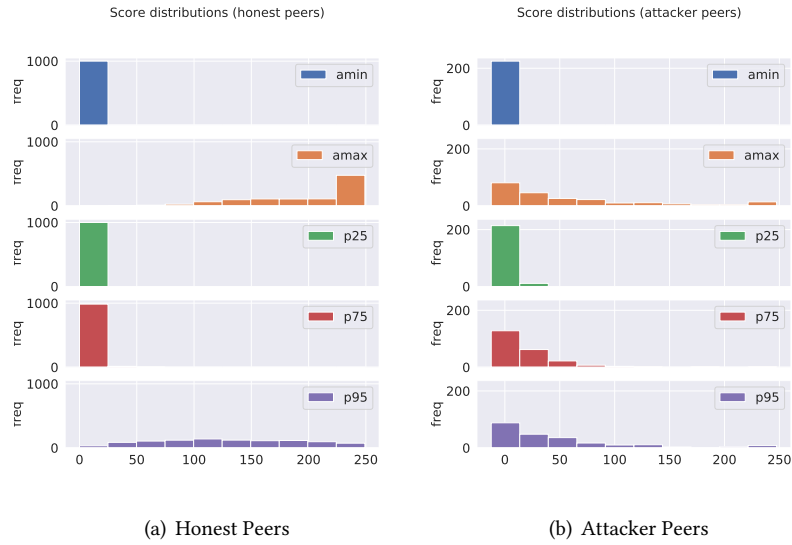


Fig. 27. Gossipsub-v1.1 Peer Score Distributions (a) Honest Peers (b) Attacker Peers

### Peer Scores - Degrade Probability 0.9

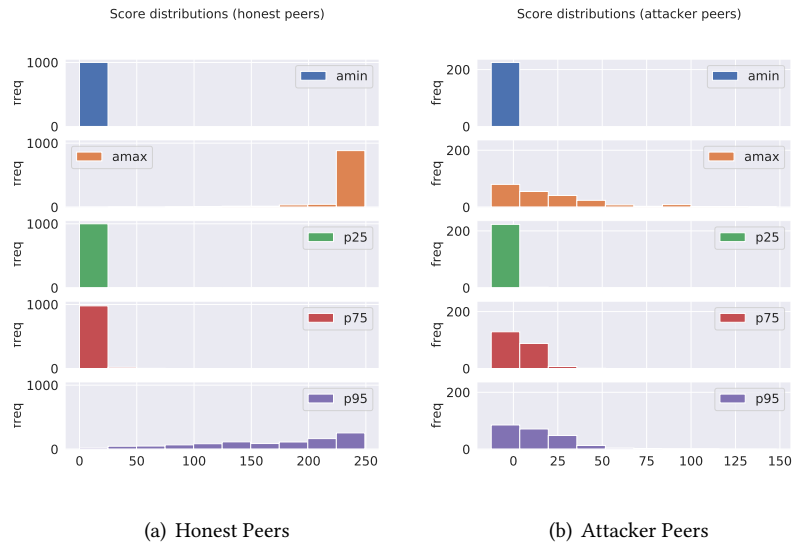


Fig. 28. Gossipsub-v1.1 Peer Score Distributions (a) Honest Peers (b) Attacker Peers

## 12 Cold Boot Attack

### 12.1 Description of Attack

This attack intends to cover the following two cases: i) the network bootstraps in a Sybil-dominated environment, and ii) new nodes are joining the network when the network is under attack. Although the first one is quite unlikely to happen, the second one is likely to occur as the network grows.

In this attack, honest and Sybil nodes join the network concurrently; honest peers attempt to build their mesh cold, while being connected to Sybils and honest peers alike. Because there is no score build up from a warm honest network to protect the mesh, the Sybils manage to largely take over the mesh initially thanks to their larger numbers.

### 12.2 Mitigation

The whole array of strategies in place for gsv1.1 comes to play in this attack scenario:

- Flood publishing and adaptive gossip dissemination ensure message propagation while the mesh is largely under Sybil control.
- Peer scoring gradually kicks out negative scoring Sybils and replaces with neutral or better scoring peers.
- Opportunistic grafting accelerates this process, patching the mesh with publishers and gossiping peers and removing squatting Sybils in order to ensure a timely mesh recovery.

### 12.3 Test Description

In this test we have 1k honest peers (100 publishers and 900 lurkers) and 4k Sybils join together at time 0. Because of their larger numbers, the Sybils take over the mesh initially, which contains 1.6 honest peers on average for each other honest peer. Publishers start publishing at the 30s mark in the test run.

Note that the opportunistic grafting period is set to 10 sec in order to get meaningful results out of this test, whereas this parameter is set to 1 min in real deployments.

## 12.4 Results

### 12.4.1 Delivery Rate.

	Gossipsub-v1.0	Gossipsub-v1.1
Peers	1000	1000
Published Messages	21700	21700
Delivered Messages	21464410	21700000
Delivery Rate	98.9%	100%

**12.4.2 Delivery Latency.** See Figs. 29 for gs-v1.0, 30 and 31 for gs-v1.1 with and without Opportunistic Grafting, and 32 and 33 for gs-v1.1 with and without Opportunistic Grafting and Gossip Propagation factor of 0.4 (as compared to the default of 0.25).

**12.4.3 Peer Score Values over Time.** See Fig. 34 and 35

**12.4.4 Peer Scores.** See Fig. 36 and 37

## 12.5 Analysis

gs-v1.0 fails to respond to the attack. It not only delivers messages with prohibitive delay, with a maximum of over 40s, but also fails to propagate some messages. gs-v1.1 doesn't lose any messages, but is initially affected by the attack with early messages delayed up to 4-5s. However it gradually recovers and ultimately achieves timely delivery for the vast majority of messages.

We present results with and without opportunistic grafting, and also include a run with the gossip factor increased to 0.4 from its default of 0.25. With the default gossip factor of 0.25, the p99 latency is 1.632 sec

### Delivery Latency Gossipsub-v1.0 - Cold Boot Attack

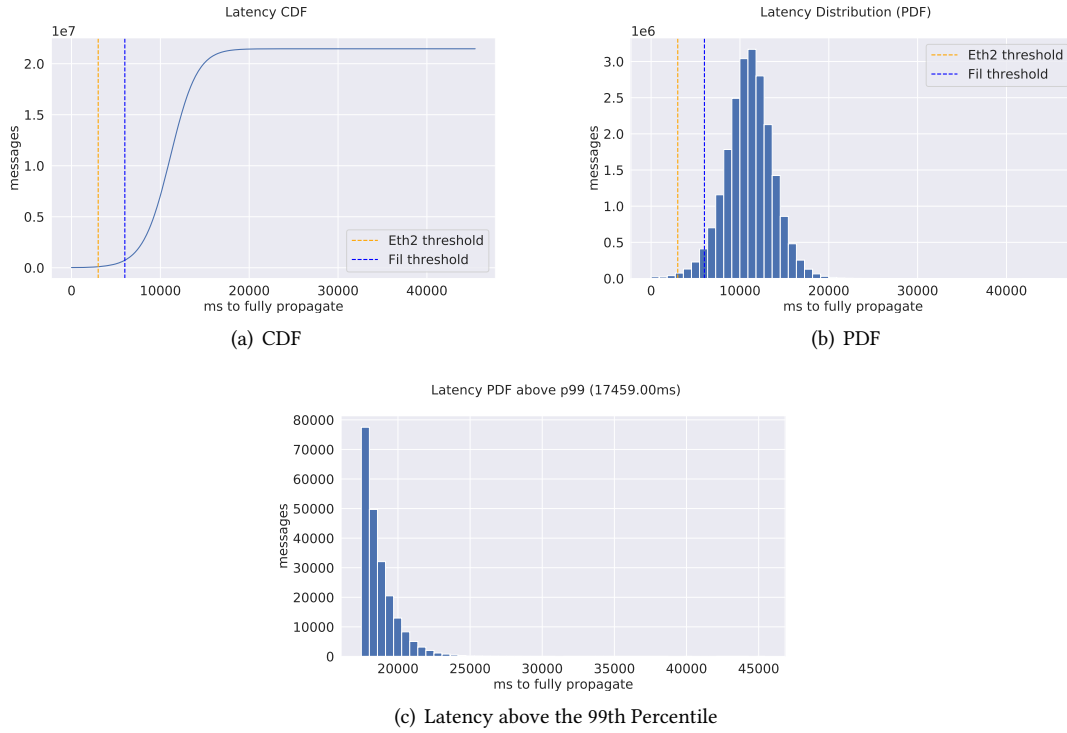


Fig. 29. Gossipsub-v1.0 (a) CDF (b) Latency above the 99th Percentile

without opportunistic grafting and 1.316 sec with opportunistic grafting. Increasing the gossip factor to 0.4 results in a p99 latency of 1.177 sec without opportunistic grafting and 1.027 with opportunistic grafting. The maximum latency also decreases with this gossip factor, reaching up to 3.5 sec. Nonetheless, increasing the gossip factor is not free as it results in 60% more gossip bandwidth.

A closer look at the latency results and the evolution of peers scores suggests that while the mesh is Sybil poisoned messages propagate through gossip, needing up to 4-5 gossip hops in some cases. Once the mesh delivery penalty kicks in, Sybils are beginning to drop out of the mesh which is recovered by honest peers. Opportunistic grafting accelerates this process as is indicated by the steeper CDF curve.

### Delivery Latency Gossipsub-v1.1 - Cold Boot Attack

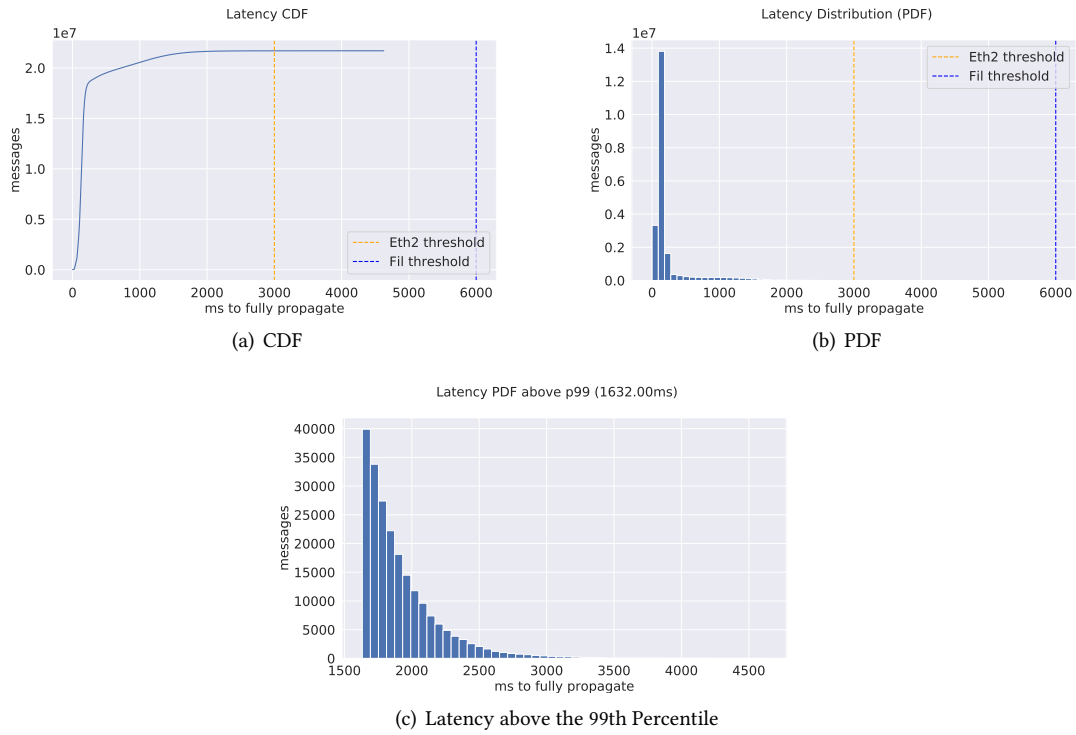


Fig. 30. Gossipsub-v1.1 Delivery Latency (a) CDF (b) PDF (c) Latency above the 99th Percentile



### Delivery Latency Gossipsub-v1.1 w/ Opportunistic Grafting - Cold Boot Attack

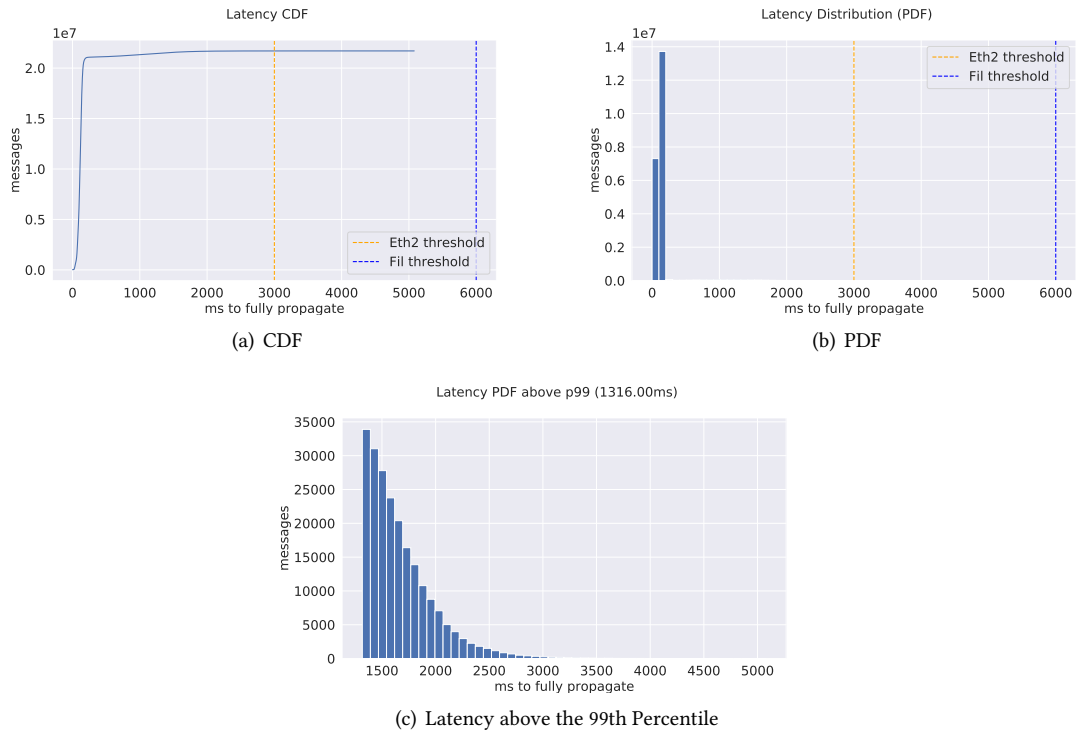


Fig. 31. Gossipsub-v1.1 Delivery Latency (a) CDF (b) PDF (c) Latency above the 99th Percentile

### Delivery Latency Gossipsub-v1.1 w/ Gossip Factor 0.4 - Cold Boot Attack

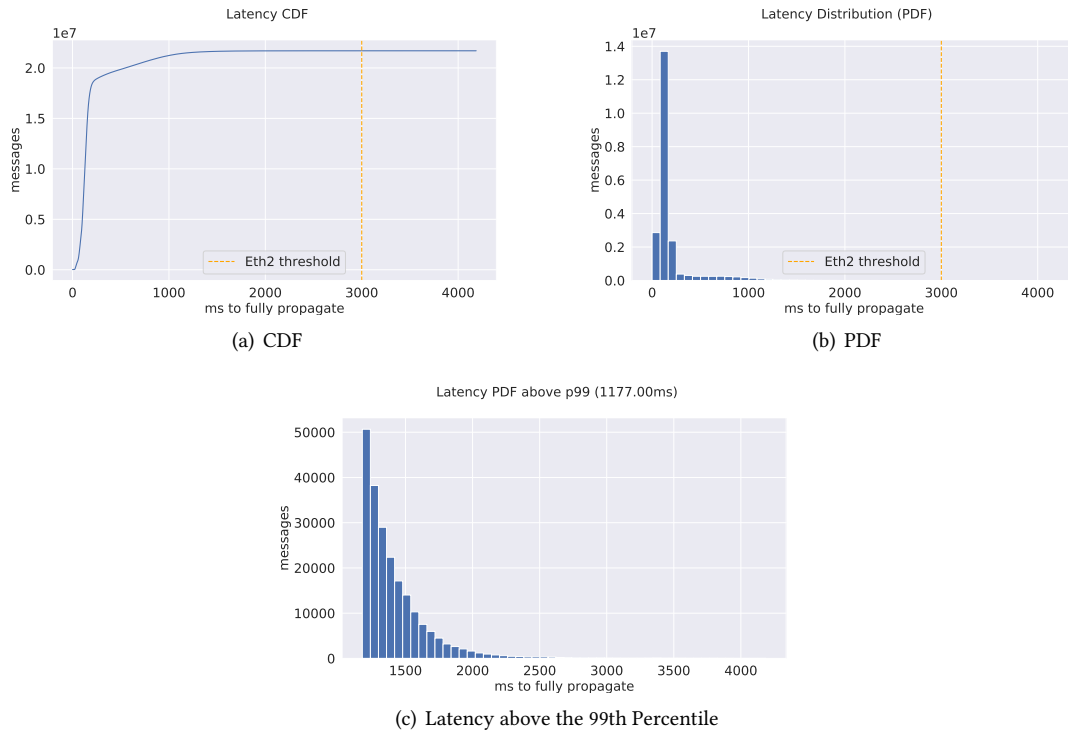


Fig. 32. Gossipsub-v1.1 Delivery Latency (a) CDF (b) PDF (c) Latency above the 99th Percentile

### Delivery Latency Gossipsub-v1.1 w/ Opportunistic Grafting & Gossip Factor 0.4 - Cold Boot Attack

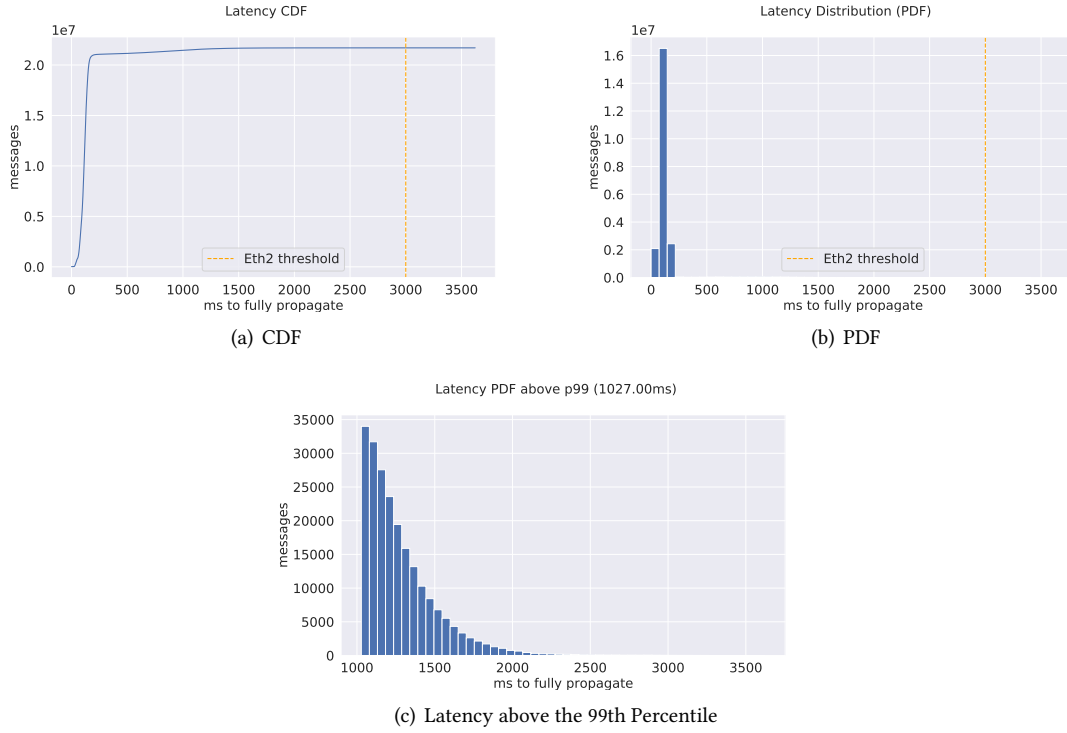


Fig. 33. Gossipsub-v1.1 Delivery Latency (a) CDF (b) PDF (c) Latency above the 99th Percentile

### Peer Score Values Over Time - Cold Boot Attack

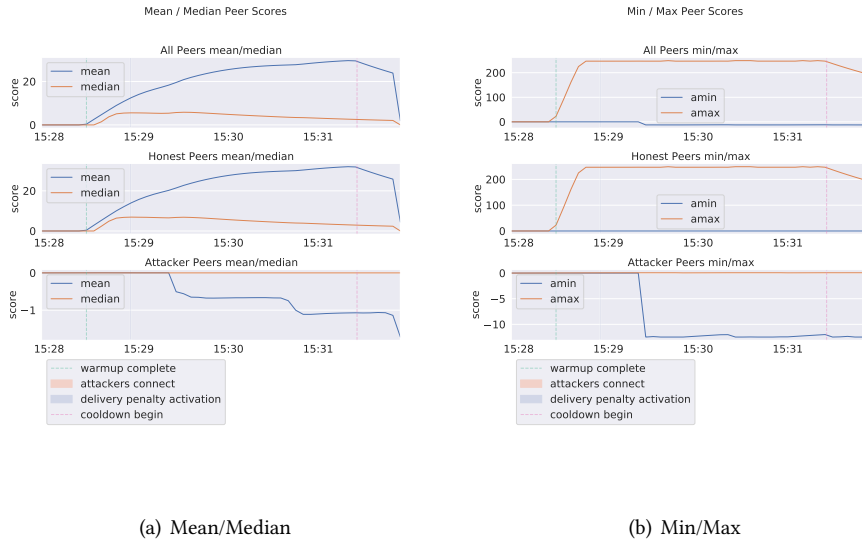


Fig. 34. Gossipsub-v1.1 Peer Score Values over Time (a) Mean/Median (b) Min/Max

### Peer Score Values Over Time w/ Opportunistic Grafting - Cold Boot Attack

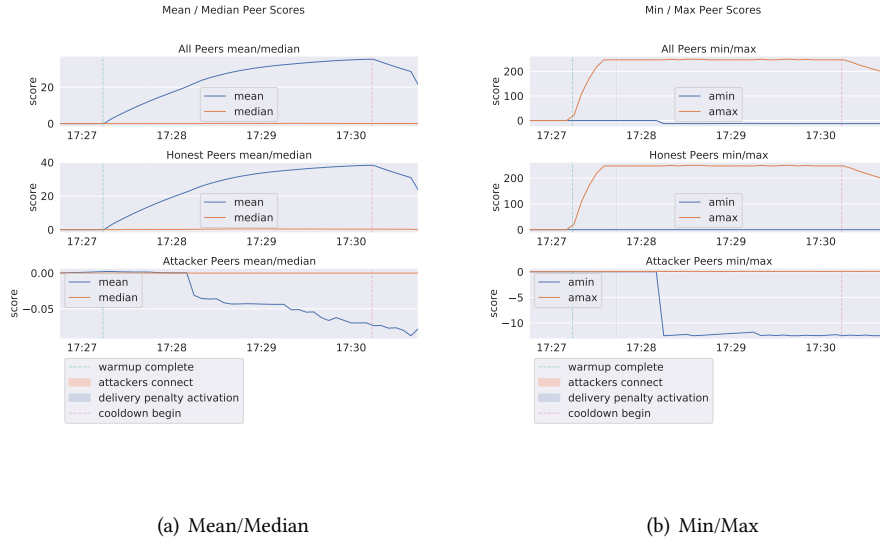


Fig. 35. Gossipsub-v1.1 Peer Score Values over Time (a) Mean/Median (b) Min/Max

### Peer Scores - Cold Boot Attack

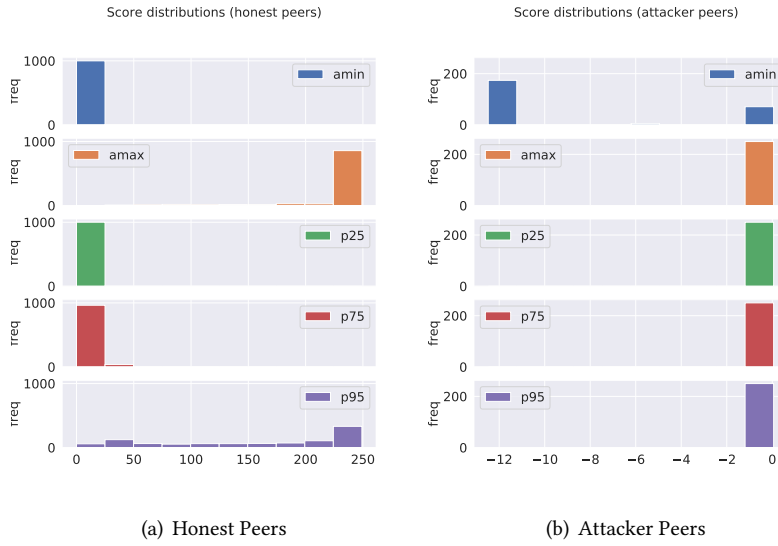


Fig. 36. Gossipsub-v1.1 Peer Score Distributions (a) Honest Peers (b) Attacker Peers

### Peer Scores w/ Opportunistic Grafting - Cold Boot Attack

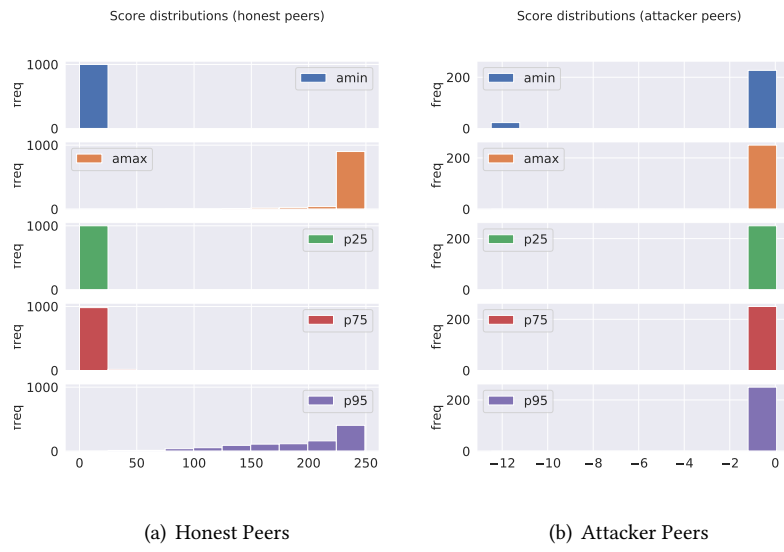


Fig. 37. Gossipsub-v1.1 Peer Score Distributions (a) Honest Peers (b) Attacker Peers

## 13 Covert Flash Attack

### 13.1 Description of Attack

This attack intends to cover cases where Sybils connect to the network and behave properly for some time in order to build up score. Then, they execute a coordinated attack whereby they stop propagating messages altogether in an attempt to completely disrupt the network.

The attack builds on the previous cold boot attack in that Sybils and honest peers connect to the network concurrently. This creates an initial mesh which is dominated by Sybils, who maintain their status by virtue of behaving properly until the time of attack.

### 13.2 Mitigation

Again, the whole array of defenses build into gsv1.1 come in to play in order to recover from the attack:

- Flood publishing and adaptive gossip dissemination ensure that messages propagate while the attack is raging on.
- Peer scoring ensures that the mesh eventually recover, by virtue of score decay.
- Opportunistic grafting accelerates the response to the attack by quickly detecting when the median score in the mesh has dropped below the threshold and grafting good scoring peers on top. This ensures that Sybils are quickly booted from the mesh once they have started their attack with publishers and gossiping honest peers grafted in their place.

### 13.3 Test Description

In this test we have 1k honest peers (100 publishers and 900 lurkers) and 4k Sybils join together at time 0, like the cold boot attack. Because of their larger numbers, the Sybils take over the mesh initially, which contains 1.6 honest peers on average for each other honest peer. Publishers start publishing at the 30s mark in the test run.

In contrast to the cold boot attack, Sybils behave properly (propagate all messages) for 2min, before they stop propagating messages altogether. The runtime of this test is 5min (longer than the other ones). Note that the opportunistic grafting period is set to 10 sec in order to get meaningful results out of this test, whereas this parameter is set to 1 min in real deployments.

### 13.4 Results

#### 13.4.1 Delivery Rate.

	Gossipsub-v1.0	Gossipsub-v1.1
Peers	1000	1000
Published Messages	36100	36100
Delivered Messages	35916578	36100000
Delivery Rate	99.4%	100%

#### 13.4.2 Delivery Latency. See Figs. 38, 39 and 40

#### 13.4.3 Peer Score Values over Time. See Fig. 41 and 42

#### 13.4.4 Peer Scores. See Fig. 43 and 44

### 13.5 Analysis

gs-v1.0 fails to respond to the attack, which results in lost messages and delivery delays of up to over 25 sec once the attack starts.

gs-v1.1 successfully responds to the attack and recovers the mesh. It doesn't lose any messages, while the maximum delivery latency during recovery tops at about 3.5s. The p99 latency is 1.619 sec without opportunistic grafting and 1.441 sec with opportunistic grafting.

### Delivery Latency Gossipsub-v1.0 - Covert Flash Attack

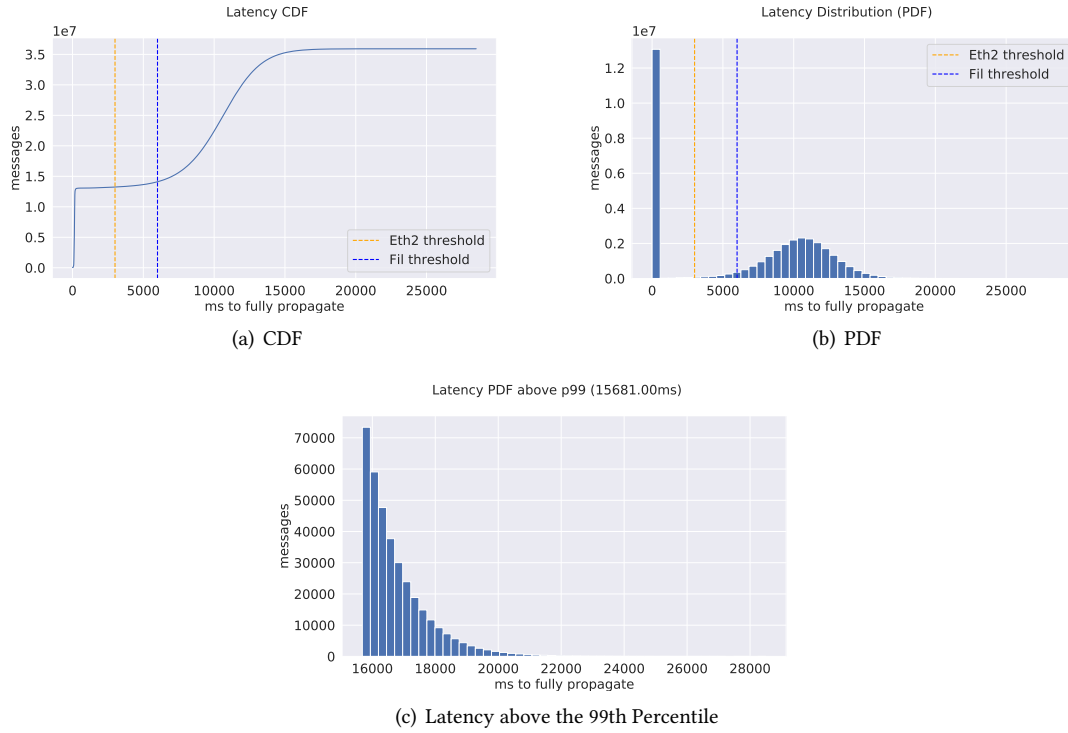


Fig. 38. Gossipsub-v1.0 (a) CDF (b) Latency above the 99th Percentile

The responsiveness of the score function responsible for the recovery is visible in the mean/median peer score over time. Initially, attackers occupy the mesh and build up a good score. Once the attack begins, the attacker scores start decreasing while the honest peer scores start increasing. The plateau in the honest peer scores with opportunistic grafting enabled (see middle plot in Fig. 44(b)) indicates that the mesh has been completely recovered within 2 min of the attack. In contrast, without opportunistic grafting (see Fig. ??(b)), the mesh does not seem to have completely recovered 3 min after the attack.

### Delivery Latency Gossipsub-v1.1 - Covert Flash Attack

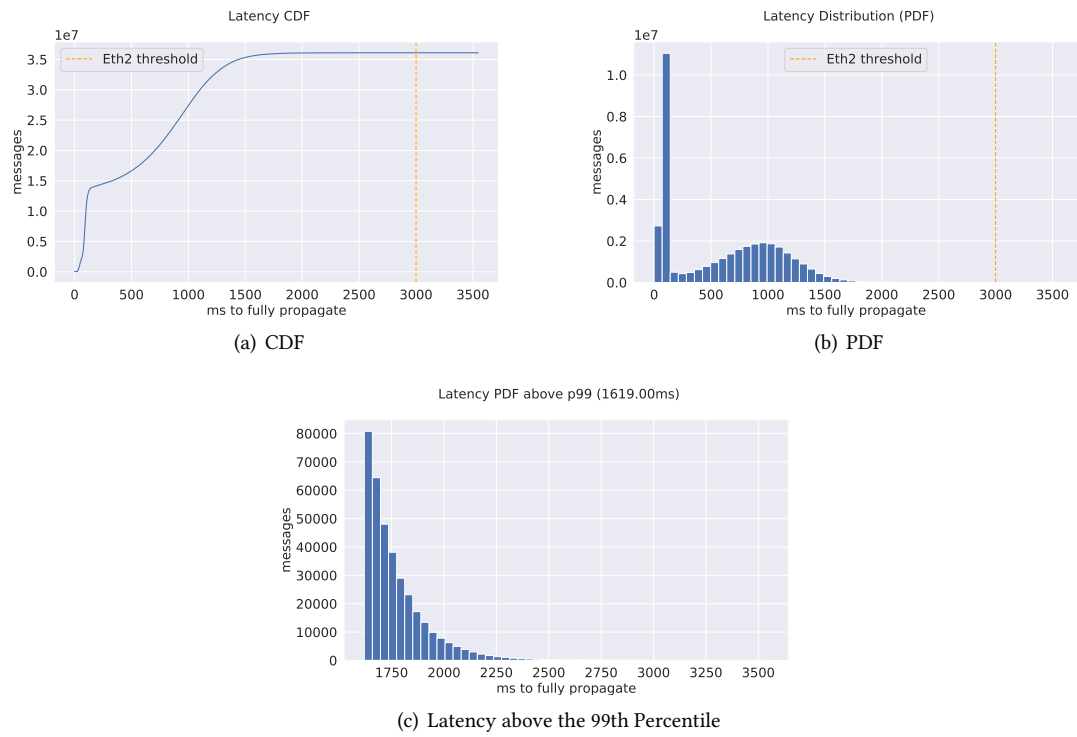


Fig. 39. Gossipsub-v1.1 Delivery Latency (a) CDF (b) PDF (c) Latency above the 99th Percentile



### Delivery Latency Gossipsub-v1.1 w/ Opportunistic Grafting - Covert Flash Attack

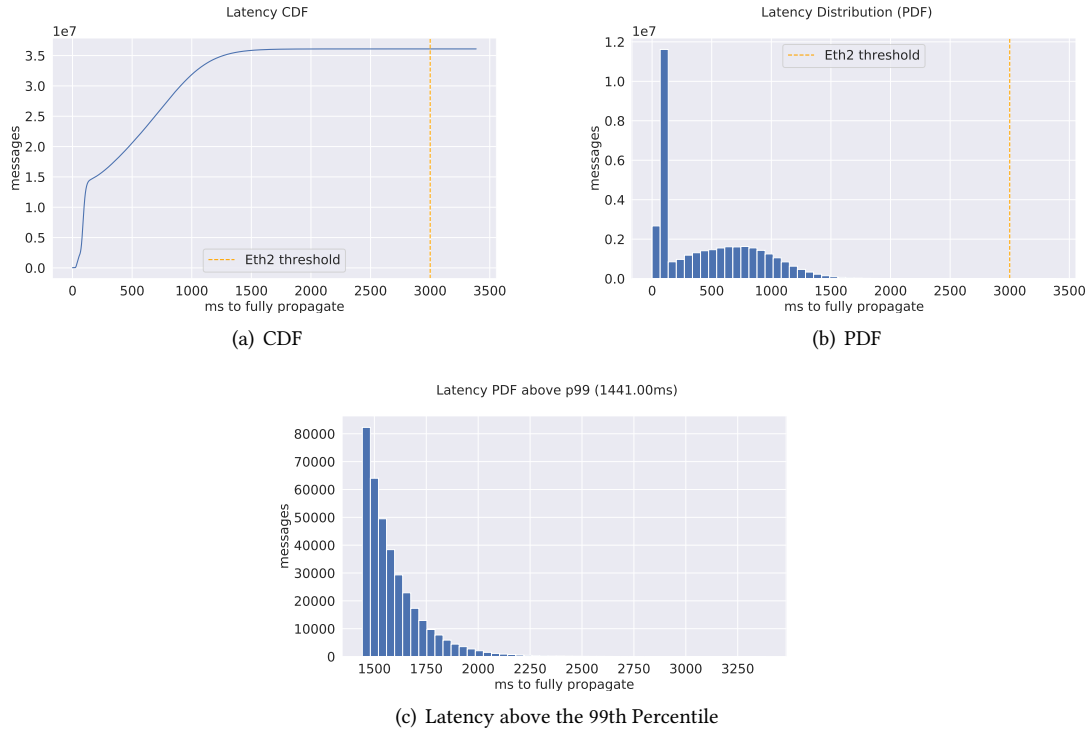


Fig. 40. Gossipsub-v1.1 Delivery Latency (a) CDF (b) PDF (c) Latency above the 99th Percentile

### Peer Score Values Over Time - Covert Flash Attack

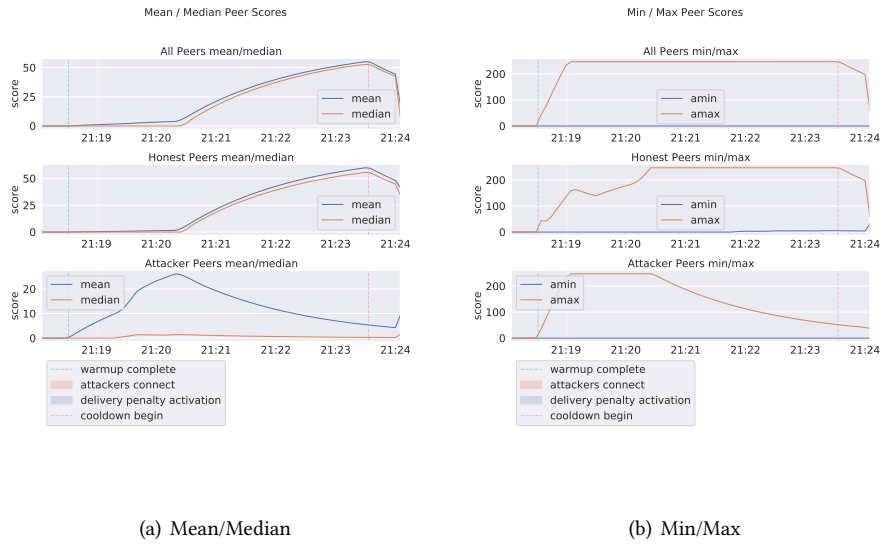


Fig. 41. Gossipsub-v1.1 Peer Score Values over Time (a) Mean/Median (b) Min/Max

### Peer Score Values Over Time w/ Opportunistic Grafting - Covert Flash Attack

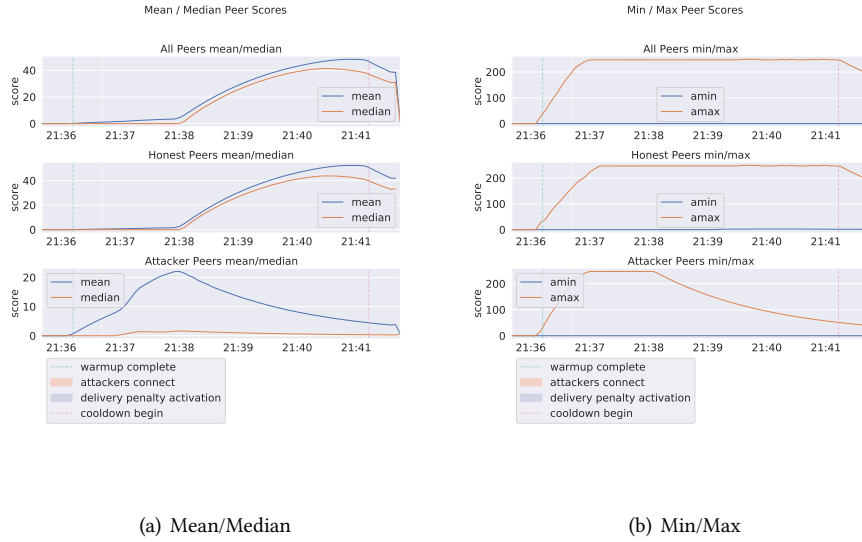


Fig. 42. Gossipsub-v1.1 Peer Score Values over Time (a) Mean/Median (b) Min/Max

### Peer Scores - Covert Flash Attack

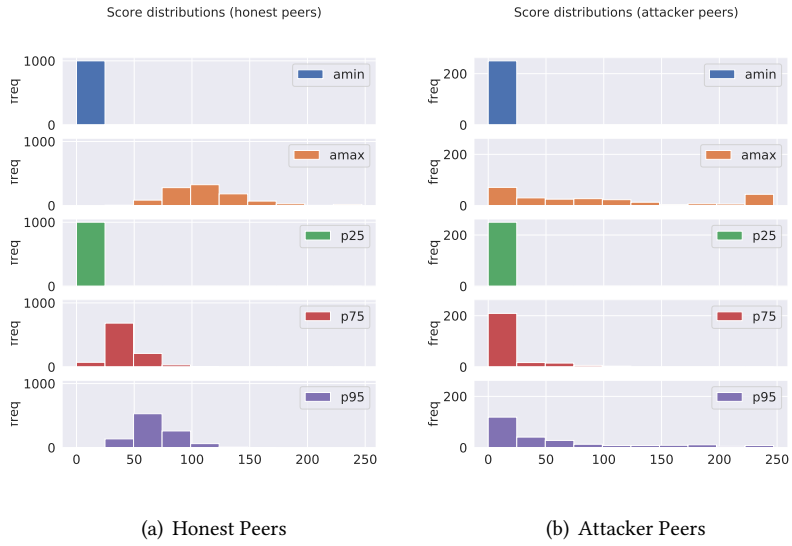


Fig. 43. Gossipsub-v1.1 Peer Score Distributions (a) Honest Peers (b) Attacker Peers

### Peer Scores w/ Opportunistic Grafting - Covert Flash Attack

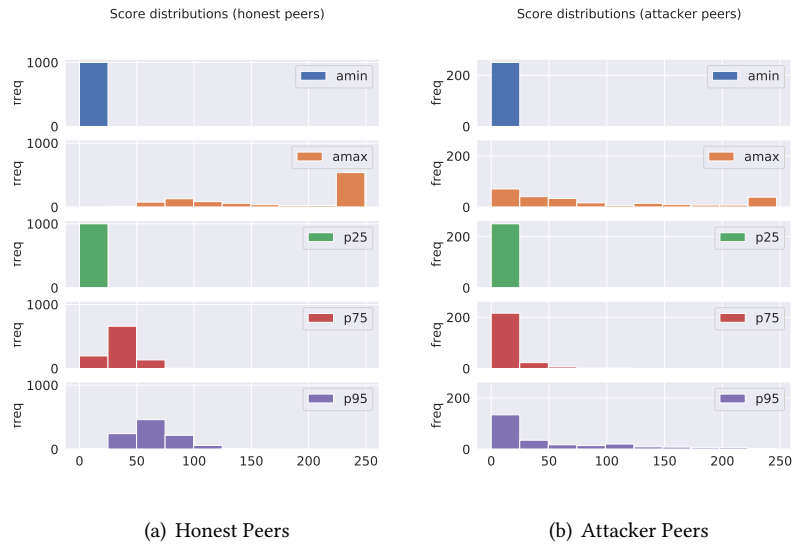


Fig. 44. Gossipsub-v1.1 Peer Score Distributions (a) Honest Peers (b) Attacker Peers

## 14 Attack at Dawn

### 14.1 Description of Attack

In this attack Sybils connect first and form an initially all Sybil mesh. Honest peers connect to each other later on and attempt to take over the mesh from the Sybils.

This attack is not realistic at bootstrap, as the genesis block cannot be created without honest peers. However, the attack may become plausible in case of a system reboot, e.g. recovering from technical or attack failures. Nonetheless we executed this test in order to stress-test and see the recovery properties of gs-v1.1.

### 14.2 Mitigation

Once again, the whole array of defenses built into gsv1.1 come into play:

- Flood publishing and adaptive gossip dissemination help messages propagate while the honest peers establish a foothold into the mesh.
- Peer scoring allows honest peers to filter out Sybils who are not propagating messages.
- Opportunistic grafting accelerates the mesh take over by quickly grafting honest publisher and gossiping peers.

### 14.3 Test Description

In this test, we have 1k honest peers (100 publishers and 900 lurkers) and 4k Sybils. In contrast to the previous tests, Sybils are first to establish connections, connecting and grafting to honest peers at time 0. Honest peers establish connections to each other after 10 sec.

Note that the opportunistic grafting period is set to 10 sec in order to get meaningful results out of this test, whereas this parameter is set to 1 min in real deployments.

## 14.4 Results

### 14.4.1 Delivery Rate.

	Gossipsub-v1.0	Gossipsub-v1.1
Peers	1000	1000
Published Messages	21700	21700
Delivered Messages	21451560	21700000
Delivery Rate	98.8%	100%

14.4.2 *Delivery Latency.* See Figs. 45, 46 and 47

14.4.3 *Peer Score Values over Time.* See Fig. 48 and 49

14.4.4 *Peer Scores.* See Fig. 50 and 51

### 14.5 Analysis

As expected, the mesh is full of attackers when honest nodes come in. When honest nodes come in, they can only occupy a few slots.

gs-v1.0 fails to recover the mesh which results in message loss and latency of up to 35s.

gs-v1.1 is successful in recovering the mesh, and even more so with opportunistic grafting enabled. All messages are delivered, with a maximum delay of about 4.5s during the recovery phase. Without opportunistic grafting the p99 latency is 1.661 sec, which drops to just 225 milliseconds with opportunistic grafting enabled.

The peer score evolution over time shows a more detailed view of the score function behaviour and how gs-v1.1 manages to recover the mesh, even without opportunistic grafting. Once the mesh delivery penalty kicks in, due to Sybil nodes dropping messages,  $P_3$  becomes negative. Upon the next heartbeat, Sybil nodes get booted from the mesh. The new peer selection now picks from the honest pool because Sybils are being backed off from their last attempt to graft. In addition,  $P_{3b}$  augments the penalty by rejecting misbehaving

### Delivery Latency Gossipsub-v1.0 - Attack at Dawn

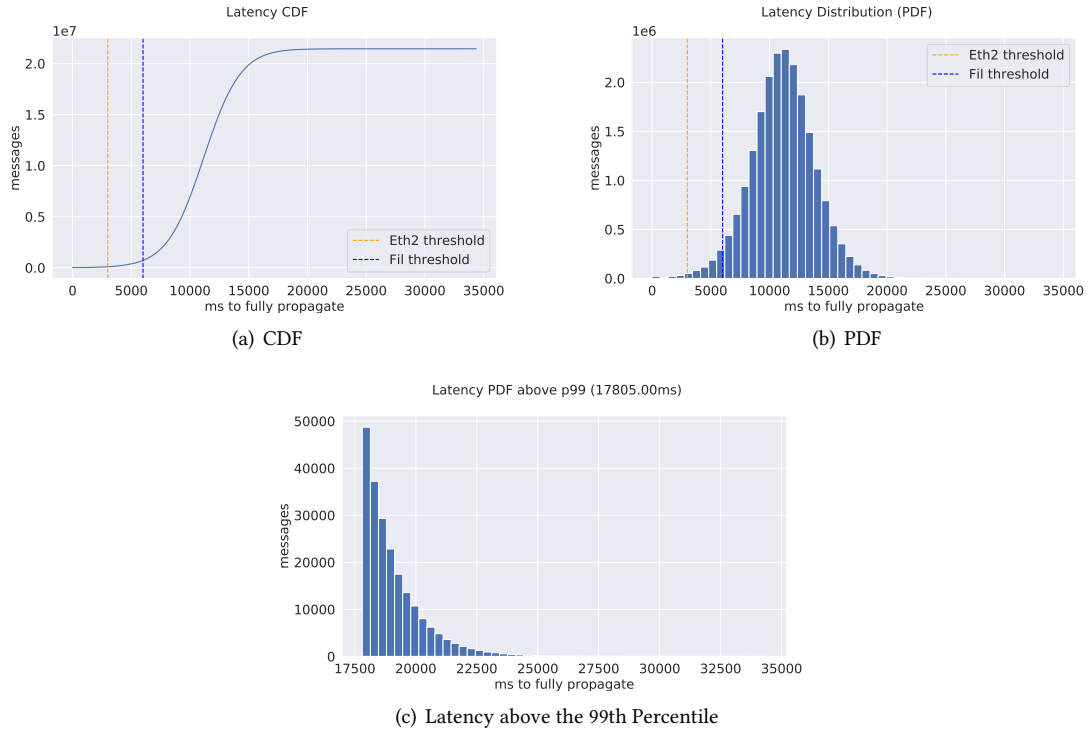


Fig. 45. Gossipsub-v1.0 (a) CDF (b) PDF (c) Latency above the 99th Percentile

nodes for longer periods of time. This results in Sybils being permanently left out of the meshes, which helps the mesh eventually recover.

With opportunistic grafting enabled the recovery time is greatly accelerated; once messages have been published or propagated through gossip, opportunistic grafting starts picking up honest peers to graft on top of Sybils in the mesh who do not propagate messages and are restricted to just the contribution of  $P_1$  in the scoring function. This quickly establishes a core of honest peers within the mesh, and results in p99 delivery times almost at the baseline.

### Delivery Latency Gossipsub-v1.1 - Attack at Dawn

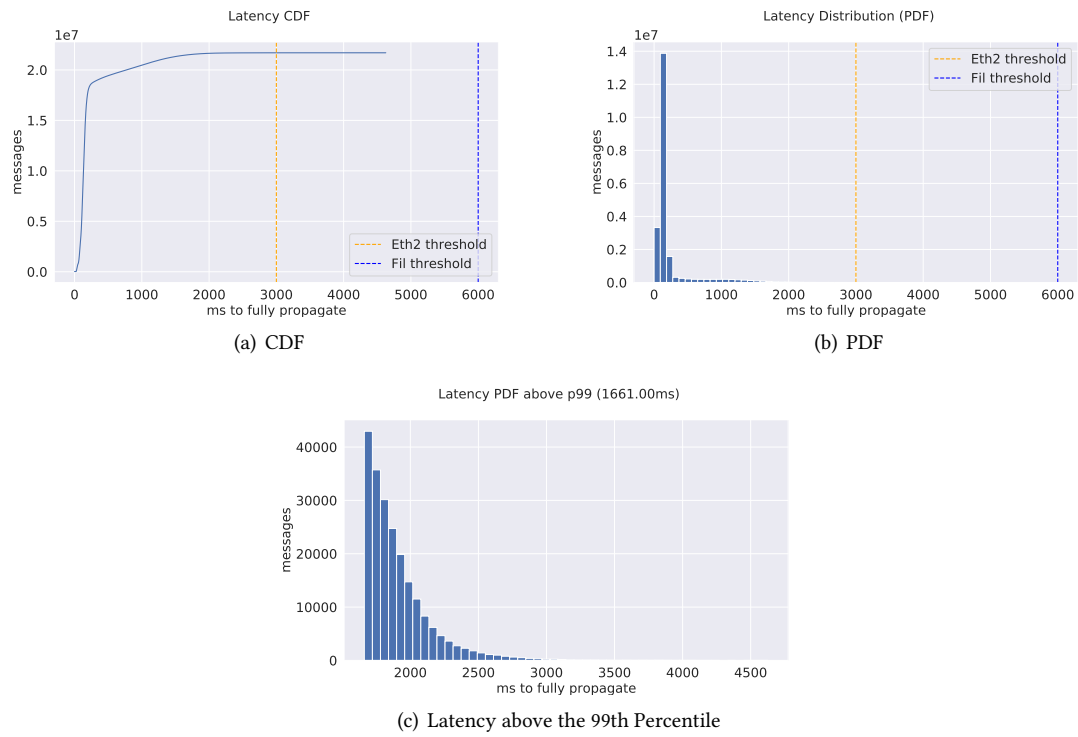


Fig. 46. Gossipsub-v1.1 Delivery Latency (a) CDF (b) PDF (c) Latency above the 99th Percentile

### Delivery Latency Gossipsub-v1.1 w/ Opportunistic Grafting - Attack at Dawn

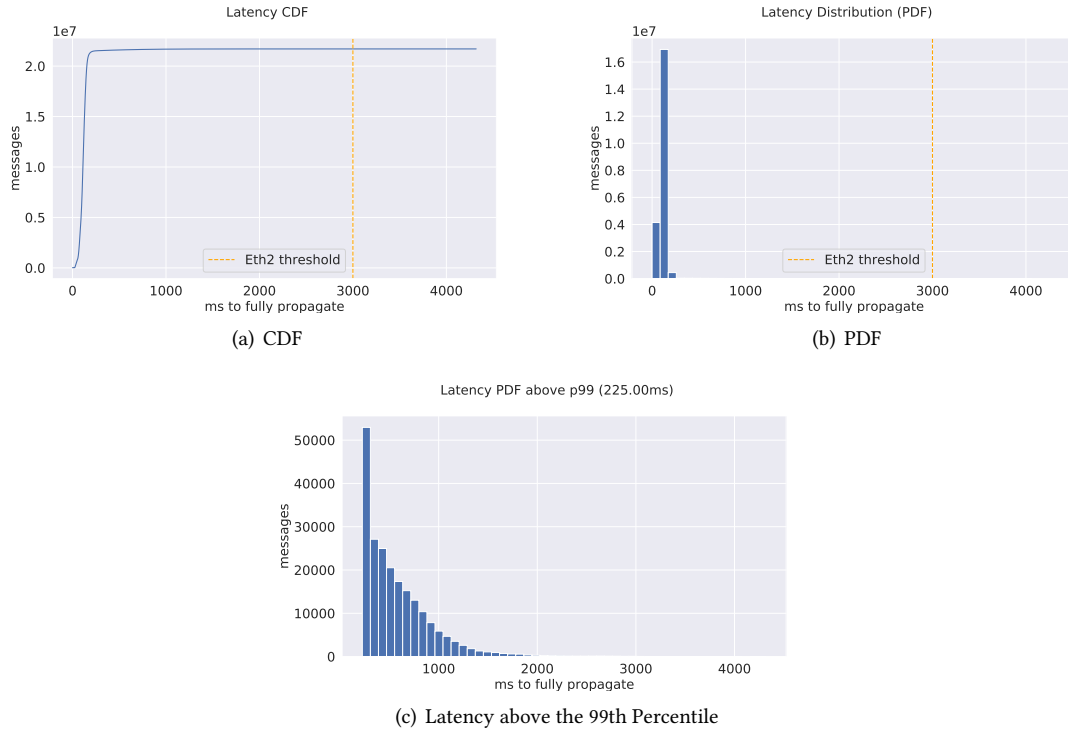


Fig. 47. Gossipsub-v1.1 Delivery Latency (a) CDF (b) PDF (c) Latency above the 99th Percentile

### Peer Score Values Over Time - Attack at Dawn

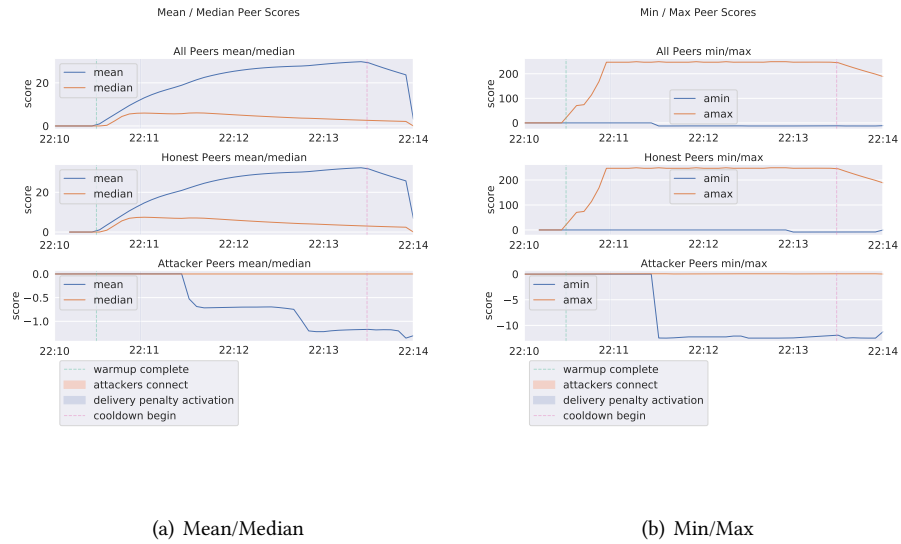


Fig. 48. Gossipsub-v1.1 Peer Score Values over Time (a) Mean/Median (b) Min/Max

### Peer Score Values Over Time w/ Opportunistic Grafting - Attack at Dawn

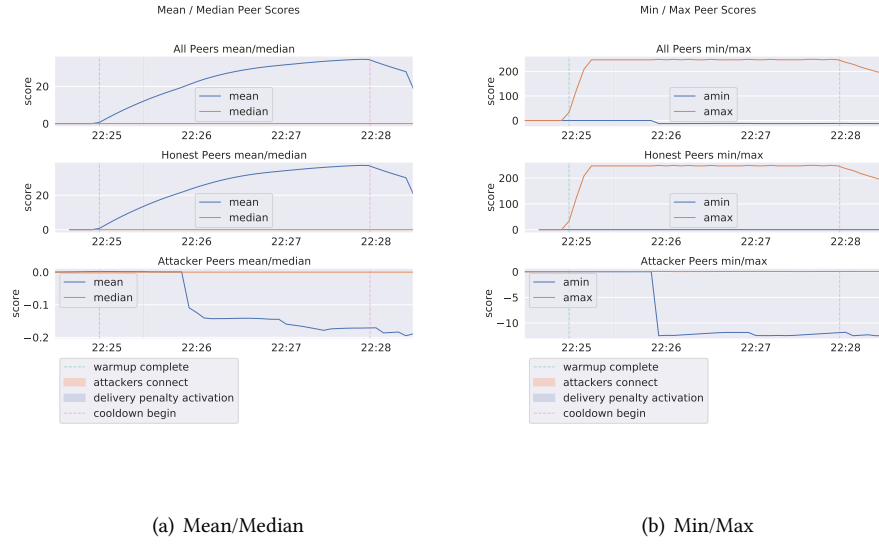


Fig. 49. Gossipsub-v1.1 Peer Score Values over Time (a) Mean/Median (b) Min/Max

### Peer Scores - Attack at Dawn

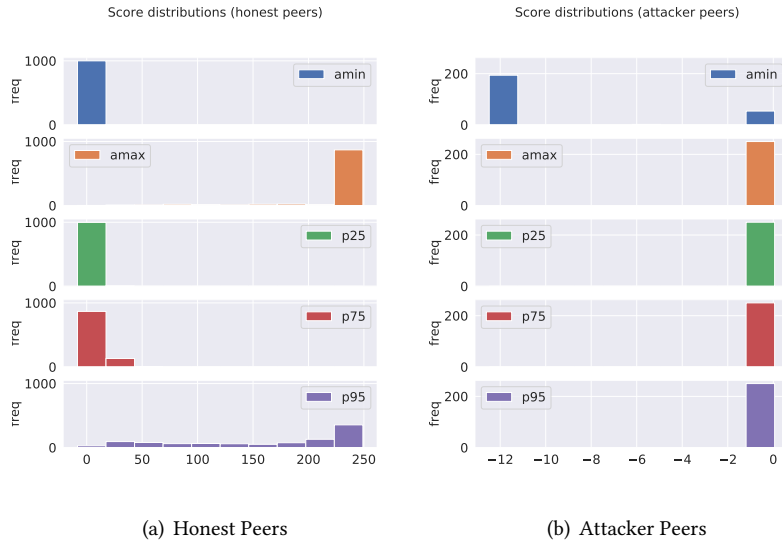


Fig. 50. Gossipsub-v1.1 Peer Score Distributions (a) Honest Peers (b) Attacker Peers



### Peer Scores w/ Opportunistic Grafting - Attack at Dawn

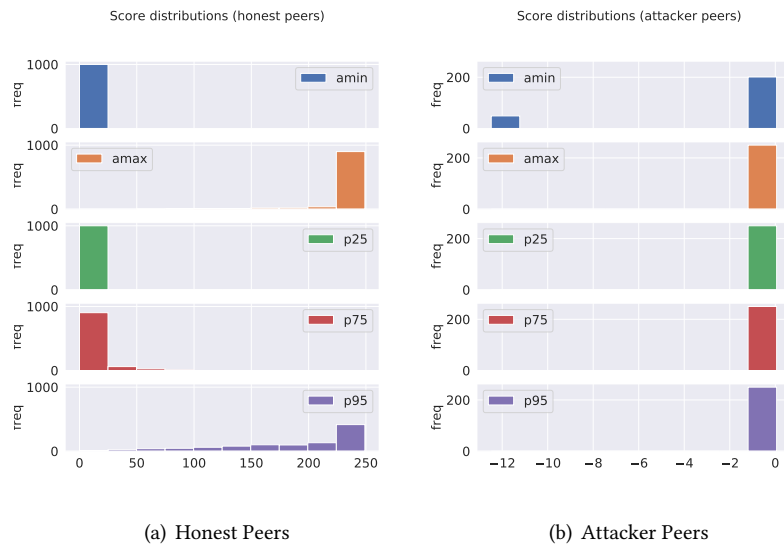


Fig. 51. Gossipsub-v1.1 Peer Score Distributions (a) Honest Peers (b) Attacker Peers

## 15 IP Collocation Attack

### 15.1 Description of Attack

This attack is the Cold Boot Attack, with the difference that all Sybils are collocated in just a few IP addresses. It is worth noting that this is not a primitive attack in itself. However, forcing attackers to obtain different IP addresses to carry out their attack effectively is necessary in order to increase the cost of the attack and therefore, its difficulty.

### 15.2 Mitigation

The  $P_6$  parameter in the scoring function is a negative cliff which increases quadratically once the number of peers collocated in the same IP address increase above the threshold (typically set to 1). This results to negative scores for all Sybils who connect to the same peer from the same IP address.

### 15.3 Test Description

In this test we have 1k honest peers (100 publishers and 900 lurkers) and 4k Sybils packed in just 25 containers. The honest peers and the Sybils connect concurrently at time 0. Publishers begin publishing messages after 30s.

Exclusively for this test, we set a value of -10 for the  $P_6$  parameter weight; the parameter was disabled in the previous tests in order to allow us to pack multiple Sybils per container and achieve high sybil:honest connection ratios.

### 15.4 Results

#### 15.4.1 Delivery Rate.

	Gossipsub-v1.0	Gossipsub-v1.1
Peers	1000	1000
Published Messages	21700	21700
Delivered Messages	21464410	21700000
Delivery Rate	98.8%	100%

#### 15.4.2 Delivery Latency. See Fig. 52

#### 15.4.3 Peer Score Values over Time. See Fig. 53

#### 15.4.4 Peer Scores. See Fig. 54

### 15.5 Analysis

The  $P_6$  parameter of the scoring function completely obliterates the attack, as most attackers obtain a negative score from the vantage point of any peer. This is illustrated in the peer score statistics and score evolution of over time. As a result, a large swath of Sybils are ignored at the time of mesh formation, which results in a largely honest peer dominated mesh. Some Sybils do try to infiltrate the mesh, but there is not a sufficient number of them to make a dent. The maximum delivery delay is at 800 ms, with the p99 at 151 ms.

### Delivery Latency Gossipsub-v1.1 - IP Collocation Attack

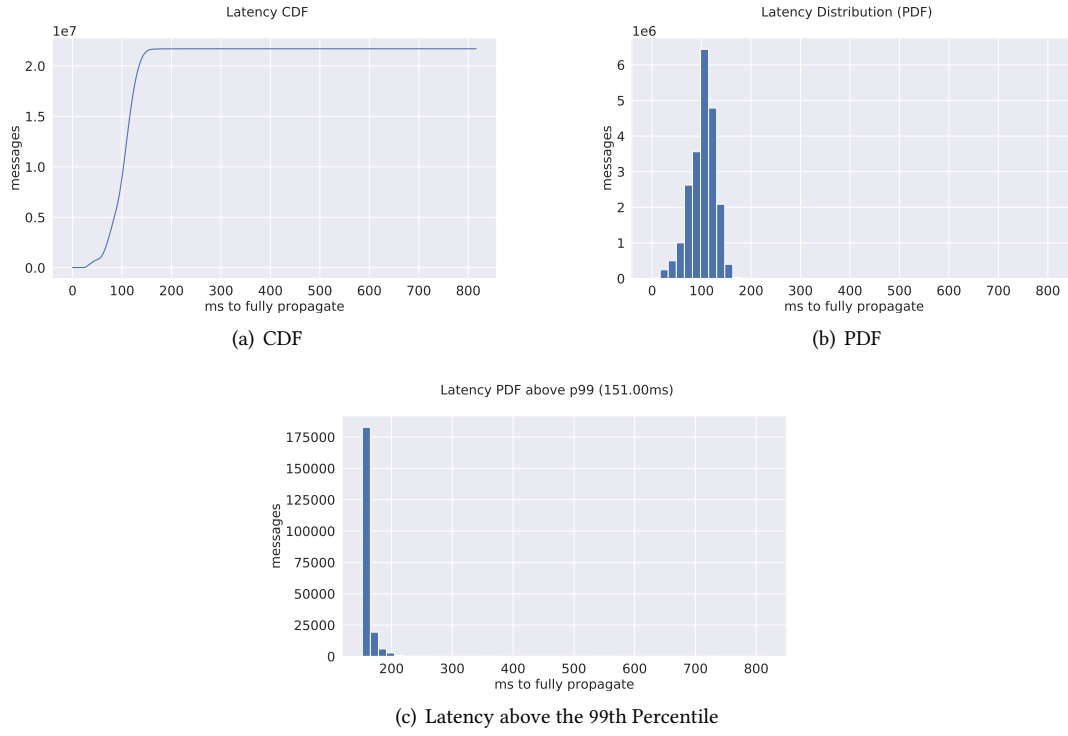


Fig. 52. Gossipsub-v1.1 Delivery Latency (a) CDF (b) PDF (c) Latency above the 99th Percentile

### Peer Score Values Over Time - IP Collocation Attack

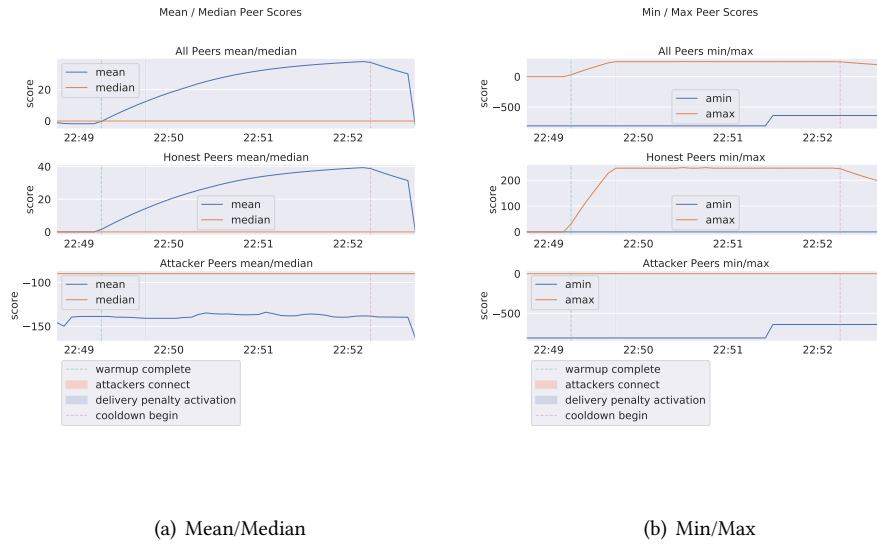


Fig. 53. Gossipsub-v1.1 Peer Score Values over Time (a) Mean/Median (b) Min/Max

Peer Scores - IP Collocation Attack

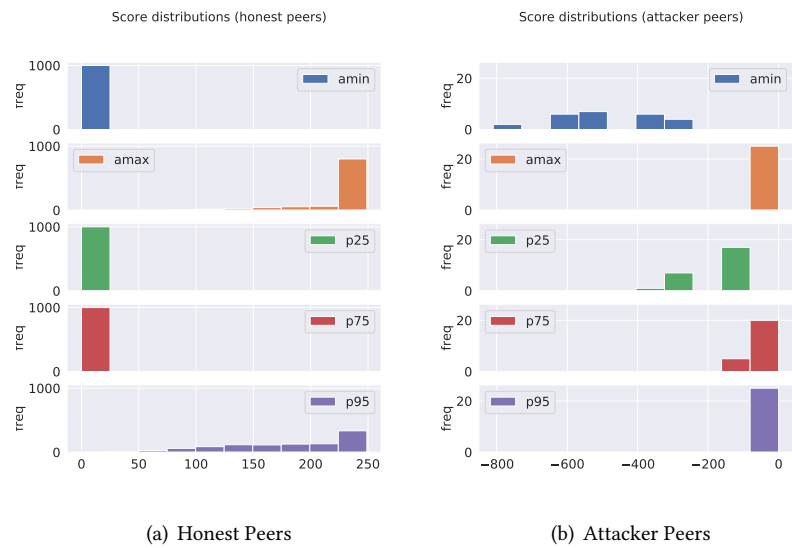


Fig. 54. Gossipsub-v1.1 Peer Score Distributions (a) Honest Peers (b) Attacker Peers