order to test our assumptions and evaluate the consequences of our proposals, we attempted to measure the effective delay function of the Mixmaster and Mixminion "supermix" over 26 days. The client software was run in its default configuration, except for Mixminion, where we forced the path length to 4 (by default it varies). To avoid our probe messages interfering which each other, we kept no more of our messages in the system than there are mixes. Latency data was collected by a Python [20] script and graphed in GNU R [21], based on a design described in [22]. This data is available for download[4].

We would, of course, have liked to evaluate the characteristic delay function of a more real-time anonymity system than Mixmaster or Mixminion, but the obvious candidate, Tor, optimises for efficiency and does not aim to protect itself from this attack. Hence, in order to demonstrate the difficulty of calculating the characteristic delay function, we have resorted to attempting to estimate it for the above systems.

5.1 Results

The distribution of measured latencies is shown in Figure 8, and the change of latency over time is shown in Figure 9, along with the distribution of latencies in two selected intervals. A statistical summary of the full data set, as well as the selected intervals, is shown in Table 1.

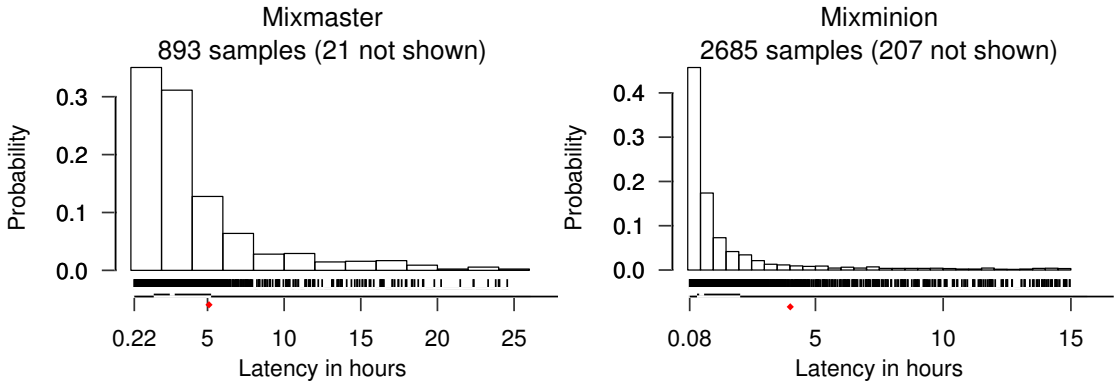

Fig. 8. Latency measurements of Mixmaster and Mixminion

As can be seen, Mixmaster has a larger median latency than Mixminion. This is because Mixminion is currently in alpha so, by default, nodes use a 10-minute timed mix. The algorithms used by Mixminion nodes at time of writing is shown in Table 2. Although the latency for most messages is below the 4×10 min limit expected for a path length of 4, 44% are above. Some of these are explained by the nodes using non-default mixing algorithms, but others are due to nodes which fail and later recover.

[4] `http://www.cl.cam.ac.uk/users/sjm217/projects/anon/`

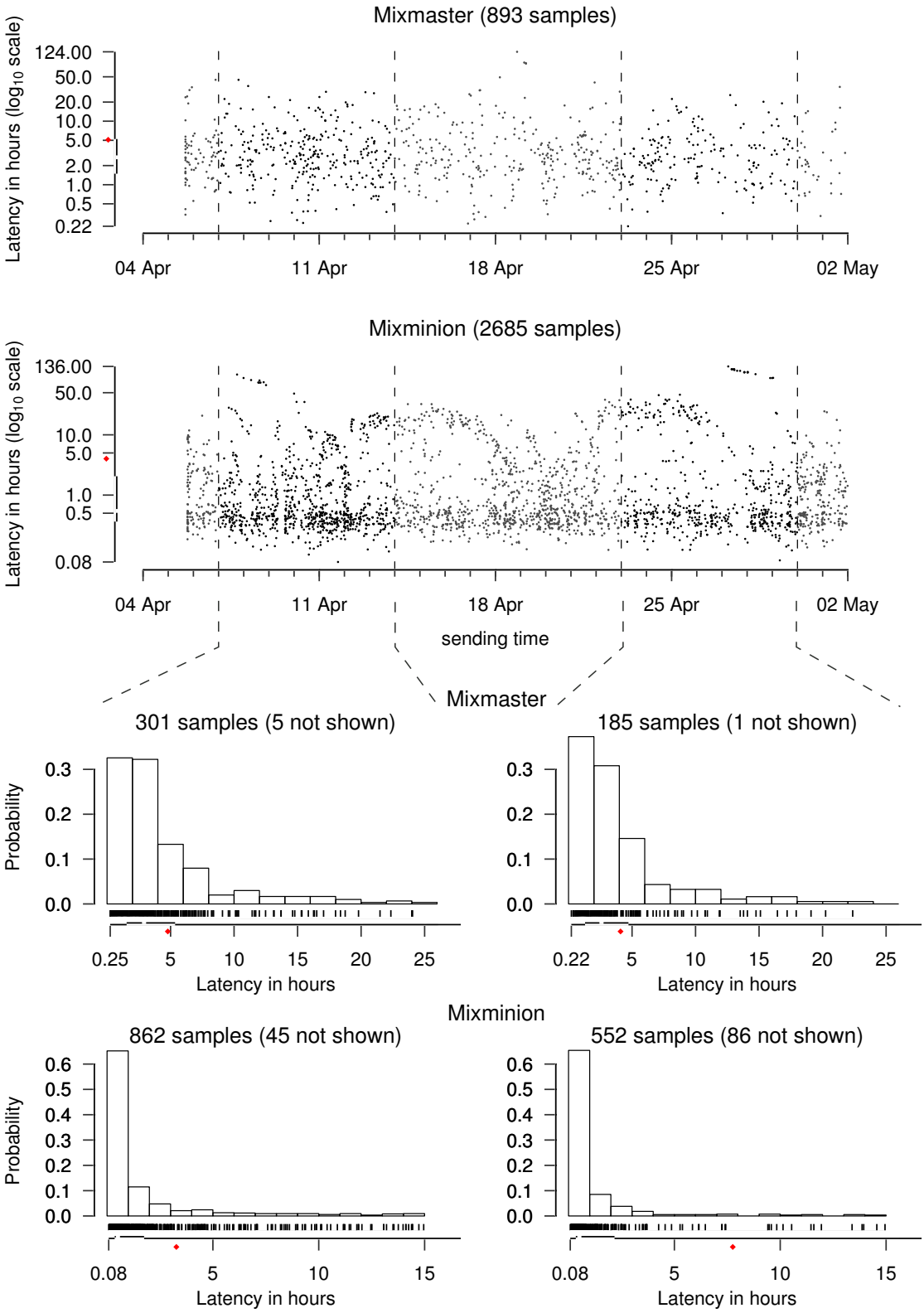

Fig. 9. Latency measurements over time

Table 1. Summary of data collected

	Mixmaster latency (hours)			Mixminion latency (hours)		
	Overall	Range 1	Range 2	Overall	Range 1	Range 2
Min.	0.22	0.25	0.22	0.08	0.08	0.08
Q.1	1.49	1.54	1.30	0.36	0.36	0.34
Med.	2.70	2.91	2.60	0.55	0.51	0.51
Q.3	5.23	5.36	4.72	2.05	1.75	2.17
Max.	123.70	44.78	25.79	136.40	100.10	136.40
Mean	5.10	4.78	4.09	4.01	3.27	7.76

Table 2. Mixminion node mixing algorithms. A timed mix flushes all messages from the pool after every mix interval. A dynamic pool, as is used in Mixmaster, flushes a randomly selected set of messages after every mix interval, such that the pool size never falls below the pool minimum size, and the percentage of the pool sent out is no more than the pool rate. A binomial dynamic pool flushes a randomly chosen number of messages, based on the number that would be flushed using the dynamic pool algorithm

Number of nodes	Mixing algorithm
25	Timed. Mix interval: 10 min (*default configuration*)
2	Timed. Mix interval: 15 min
1	Timed. Mix interval: 20 min
1	Timed. Mix interval: 30 min
1	Dynamic pool. Mix interval: 30 min, Pool Rate: 50%, Pool Minimum Size: 5
1	Binomial dynamic pool. Mix interval: 10 min, Pool Rate: 70%, Pool Minimum Size: 3
1	Binomial dynamic pool. Mix interval: 30 min, Pool Rate: 50%, Pool Minimum Size: 5