**Team 17 Results, Anonychat**

Ben Korza, Mark Fitzgibbon, Mitchell Caisse

Results Section

The main goal of our project was to create an Anonymous chat program to allow users to communicate with each other anonymously. We have defined several metrics that determine how successful we were in creating this protocol. These metrics include whether an adversary could determine who sent a packet based on the time difference between a successful and unsuccessful encryption. We also tested latency to confirm our protocol was useful in practice. Finally, we tested our protocol’s network utilization to ensure the it does not saturate the network.

To ensure the use of encryption wouldn’t add significant overhead to our protocol, we tested the time it takes for messages of various sizes to be encrypted and then decrypted. We sent messages with lengths increasing in increments of 20, and recorded the times of both successful and unsuccessful decryptions. The results are shown in Graph A.

Graph A

These results suggest that the use of encryption did not add significant overhead to our protocol. In addition, the difference between the successful and unsuccessful decryption times is both small and random enough such that we are confident an adversary could not confidently determine who sent a message based on these times.

Another thing crucial to the success of our protocol was its bandwidth consumption. In order to make sure Anonychat was scalable we tested our network utilization. We did this by running the application on 25 clients running locally over the course of five minutes. During this time, messages were sent between clients at random intervals ranging from five to 35 seconds. As this happened, we measured the amount of packets that existed in the network. The results of these tests are reflected in Graph B.

Graph B

From these results we can gather some information.

Since we desired Anonychat to be a practical protocol, it was crucial we test the latency of sending packets across the network. To do this we created three nodes and had them communicate with each other. We recorded the RTTs of packets, and the results can be seen in the following three tables.

Node 1 RTTs

Sending RTT

Node (ms)

|  |  |
| --- | --- |
| 2 | 13.561035 |
| 2 | 12.787842 |
| 2 | 13.983887 |
| 3 | 7.923887 |
| 3 | 10.42920 |
| 2 | 10.164062 |
| 3 | 10.879150 |
| 2 | 13.978530 |
| 2 | 13.126953 |
| 2 | 13.251221 |
| 3 | 5.745850 |
| 2 | 11.463867 |
| 2 | 13.580078 |
| 3 | 8.060059 |
| 2 | 10.706299 |
| 3 | 3.977051 |

Node 2 RTTs

Sending RTT

Node (ms)

|  |  |
| --- | --- |
| 3 | 8.164062 |
| 3 | 7.083740 |
| 3 | 13.013672 |
| 1 | 13.936035 |
| 3 | 6.549805 |
| 1 | 12.625977 |
| 1 | 14.530029 |
| 3 | 7.388184 |
| 3 | 8.044922 |
| 1 | 12.937012 |
| 3 | 6.104004 |
| 3 | 6.746094 |
| 1 | 13.744141 |
| 1 | 13.084961 |

Node 3 RTTs

Sending RTT

Node (ms)

|  |  |
| --- | --- |
| 1 | 41.361084 |
| 2 | 31.485107 |
| 1 | 40.700928 |
| 2 | 31.006104 |
| 1 | 32.227051 |
| 2 | 28.937988 |
| 1 | 37.948975 |
| 2 | 27.944092 |
| 1 | 37.994873 |
| 2 | 111.469971 |
| 1 | 45.537109 |
| 2 | 33.472168 |

From this data we conclude something.

During this test the ANS reshuffled the peer list x times. These reshuffles took y ms on average.

The last thing we needed to check was encryption. Using wireshark, we confirmed the messages were indeed encrypted. No identical packets were found either.