# Discv5 Research Project Simulation Results

**Default parameters**: (defined in final.cfg)

- Network size: 5000
- Queue size per topic in topic table: 50 registrations/topic
- Topic table capacity: 100
- Ticket table bucket size: 3
- Search table bucket size: 16
- Search strategy: Random-Bucket
- Registration strategy: No-Spam (i.e., Removing Ticket after expiration)
- 1 hour run
- Results limit lookup: 50.
- Turbulence events: each 1.5 sec.
- Registration lifetime (i.e., expire after): 5 minutes

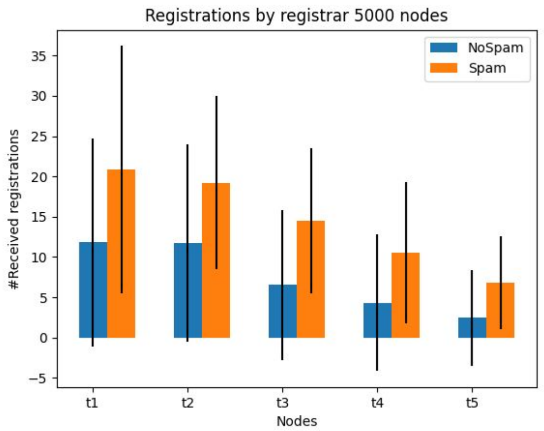
## Design goals:

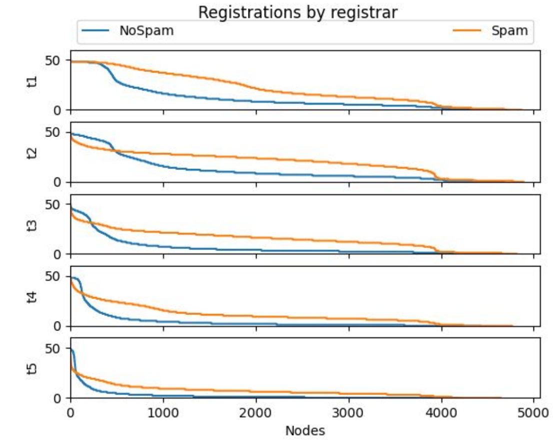
Under the assumption listed above, we target to achieve the following goals:

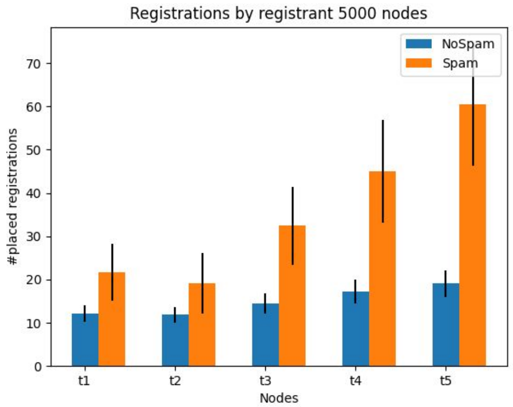
- G1 - all the registrants (regardless of the topic they register for) should be able to place their advertisements in the network; that is, no registrants can be globally denied registrations.
- G2 - all the registrants within each topic should have a similar probability of being discovered by their peers.
- G3 - the load (in terms of sent and received messages) should be equally distributed across all the nodes regardless of their ID and location in the network
- G4 - the registration operation should be efficient in terms of time (fast) for all the registrants
- G5 - the registration operation should be efficient in terms of overhead (low amount of sent/received messages) for all the registrants.
- G6 - the lookup operation should be efficient in terms of time (fast) and messages sent (hop count) for all the query nodes.
- G7 - the number of total registrations per topic should be proportional to the popularity of the topic (the number of registrants).
- G8 - the protocol should be resistant to network dynamics (nodes joining leaving)
- G9 - the protocol should be resistant to attacks launched by malicious nodes.

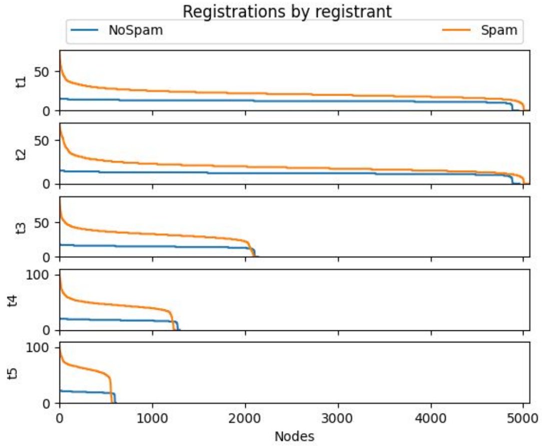
**Results per goal:**

**Goal 1**: all the registrants (regardless of the topic they register for) should be able to place their advertisements in the network. This means, no registrants should be globally denied registrations.





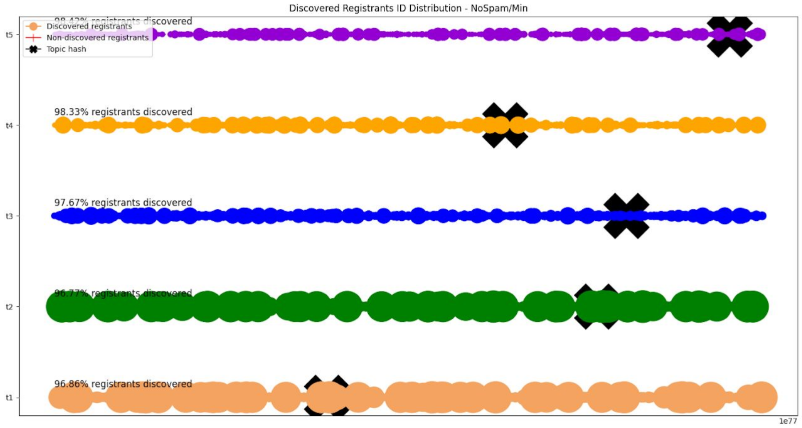


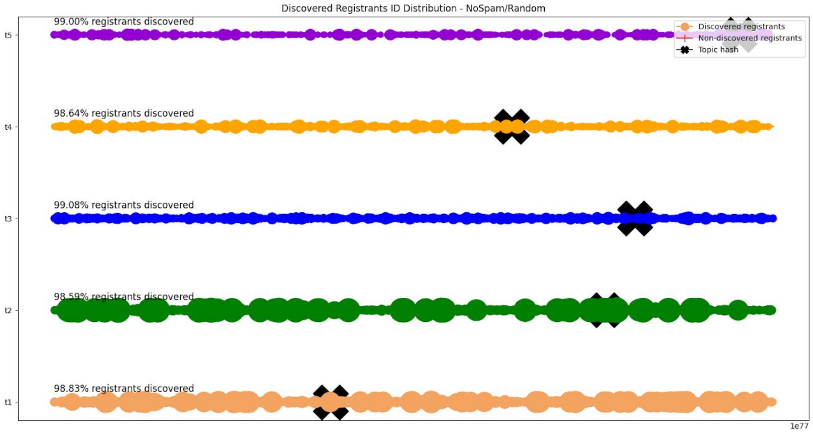


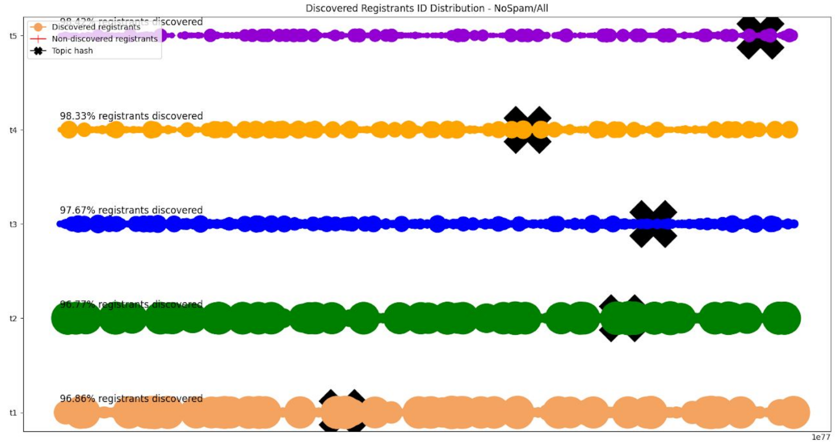
- What we show:
  - Which nodes place registrations for a certain topic and the amount distribution
  - We compare two approaches: Removing Ticket after Registration (Spam) or Removing Ticket after expiration (No Spam)

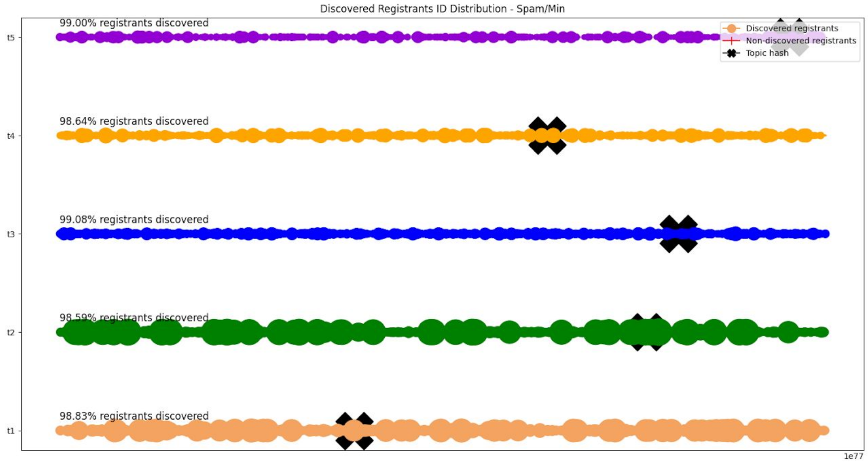
- What we observe:
  - All nodes have placed registrations, and therefore are discoverable and reachable.
  - Those nodes that register for less popular topics, place more registrations than popular topics, which have to compete for the storage available.
  - Registrars close to topic hash receive more registrations.

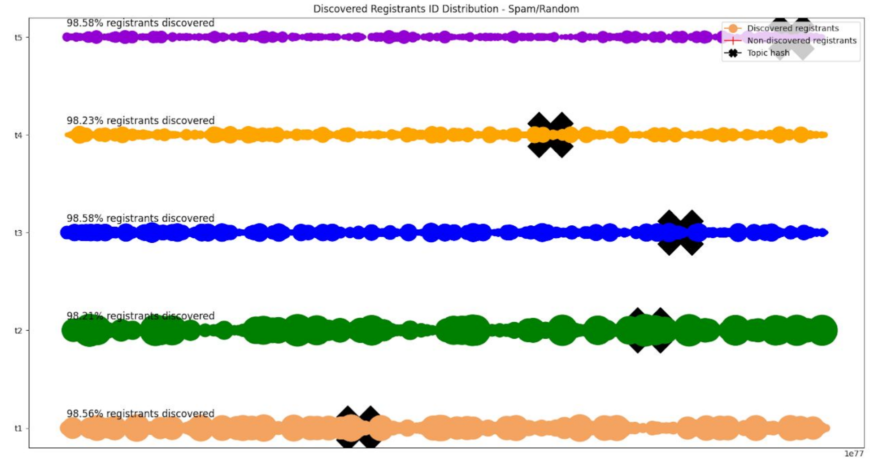
**Goal 2:** all the registrants within each topic should have a similar probability of being discovered by their peers.

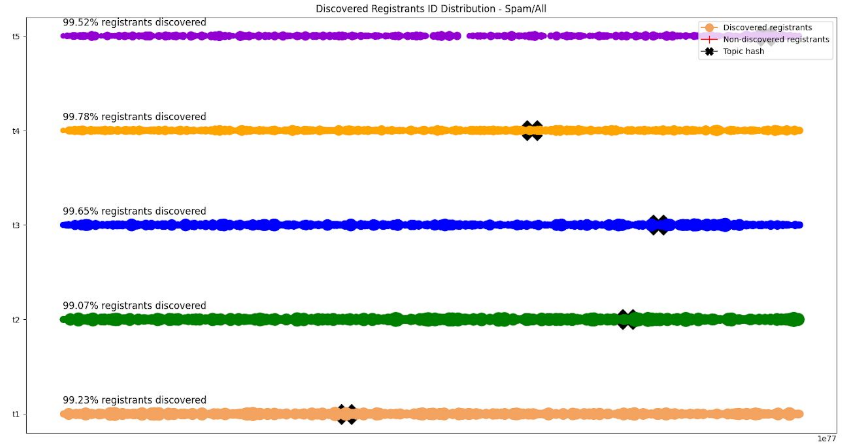





Discovered Registrants ID Distribution - NoSpam/All
98.42% registrants discovered
98.33% registrants discovered
97.67% registrants discovered
96.77% registrants discovered
96.86% registrants discovered
Discovered registrants
Non-discovered registrants
Topic hash
t5
t4
t3
t2
t1
1e77


Discovered Registrants ID Distribution - Spam/Min
99.00% registrants discovered
98.64% registrants discovered
99.08% registrants discovered
98.59% registrants discovered
98.83% registrants discovered
Discovered registrants
Non-discovered registrants
Topic hash
t5
t4
t3
t2
t1
1e77

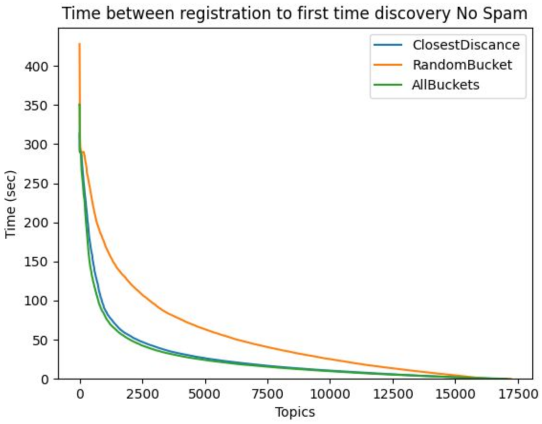




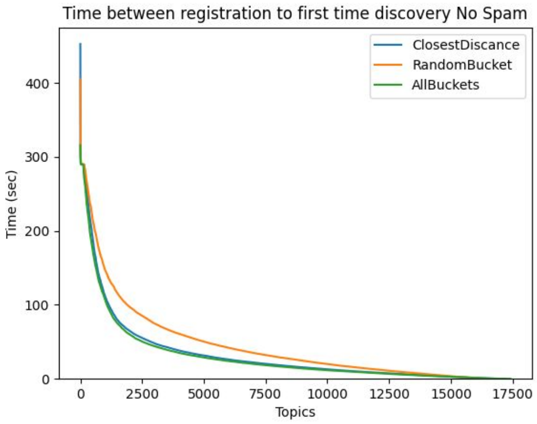
- What we show:
  - Which nodes are discovered for a certain topic
  - We compare two approaches: Removing Ticket after Registration (Spam) or Removing Ticket after lifetime (No Spam) and all search strategies: Closest distance, random bucket, all buckets.
  - Elapsed time (discovery time) from the time a node registers until it is discovered.

- What we observe:
  - All nodes have placed registrations, and therefore are discoverable and reachable.
  - Most of them are discovered during simulation (~99%)
  - All buckets lookup strategy seems to be the best in terms of discovered nodes distribution.

- Random bucket lookup strategy is slightly slower than the rest in terms of discovery time.

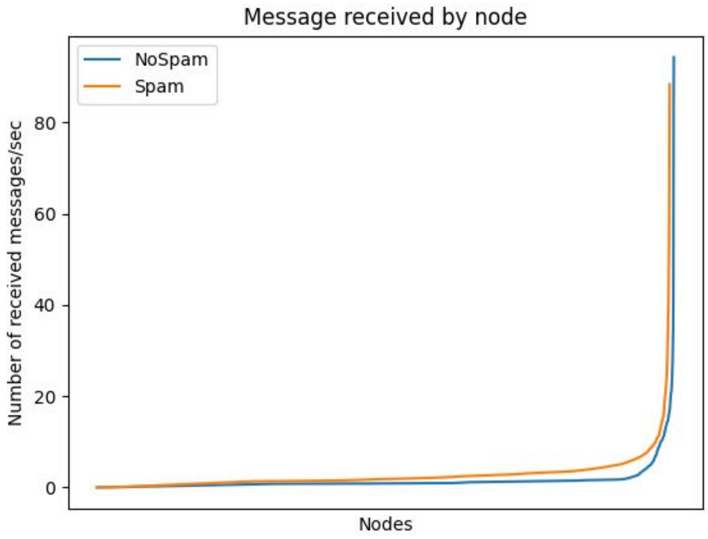


a) No-spam



b) Spam

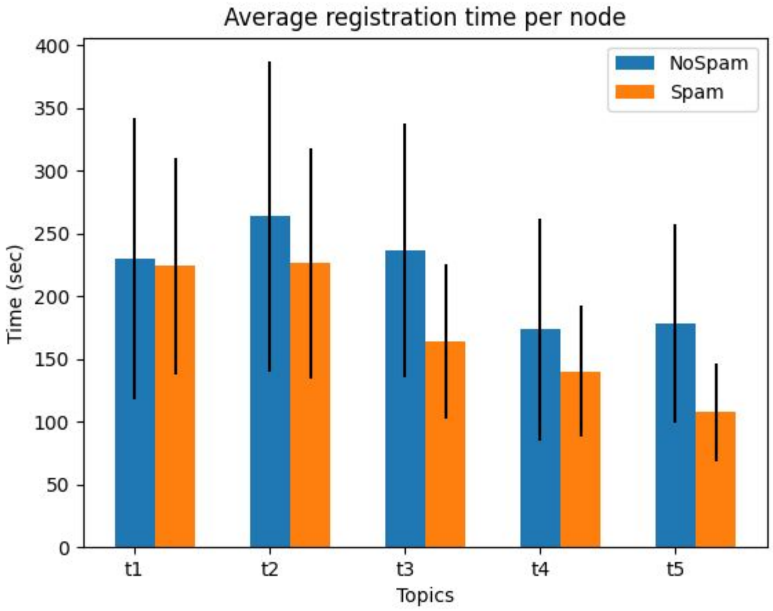
**Goal 3**: the load (in terms of sent and received messages) should be equally distributed across all the nodes regardless of their ID and location in the network



- What we show:
  - The distribution of the number of messages received per node.
  - We compare two approaches: Removing Ticket after Registration (Spam) or Removing Ticket after expiry (No Spam).

- What we observe:
  - We observe there are some nodes in the network (nodes close to the topic hash id) that receive much more density of messages than other nodes.
  - However, due to the backlog mechanism using tickets, these messages are bound to a certain limit since with the increase of tickets, waiting times are going to be increased and new messages will be autoregulated.

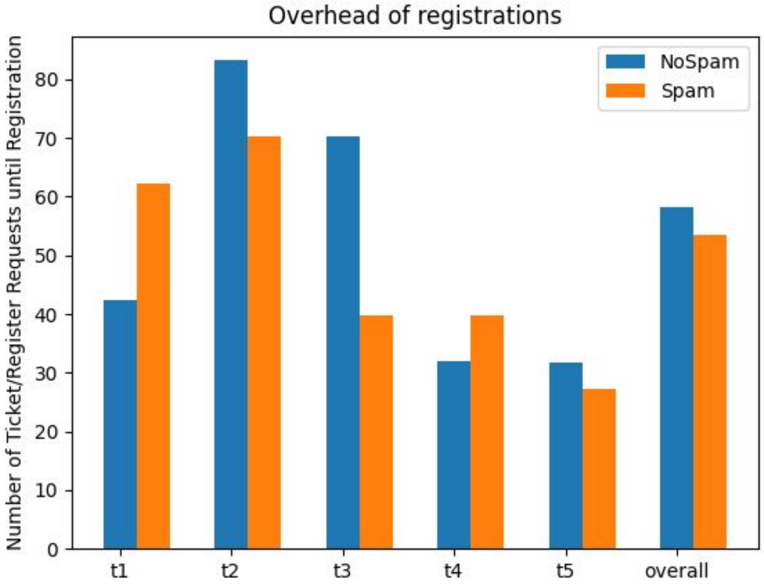
**Goal 4**: the registration operation should be efficient in terms of time (fast) for all the registrants



- What we show:
  - The average time required for a node advertising a topic to place a registration in another node.
  - We compare two approaches: Removing Ticket after Registration (Spam) or Removing Ticket after expiry (No Spam)

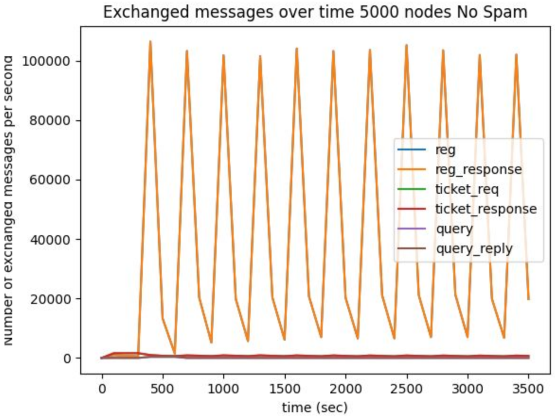
- What we observe:
  - In these graphs we observe that for popular topics it is required more time to register, since the occupancy of advertising space tends to be higher.
  - However, we observe the difference is not substantial and is possible to place registrations for all topics without having to wait more than 5 minutes.
  - Also, using Spam method the time required for registration is higher due to higher occupancy, although the differences are not important.

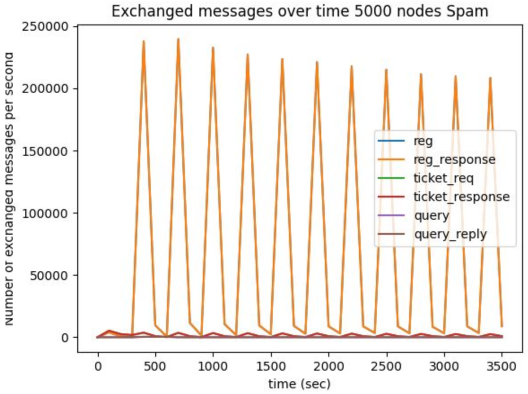
**Goal 5**: the registration operation should be efficient in terms of overhead (low amount of sent/received messages) for all the registrants



- What we show:
  - Messages required per successful registration
  - We compare two approaches: Removing Ticket after Registration (Spam) or Removing Ticket after expiry (No Spam)

- What we observe:
  - We observe that, on average, it requires a high number of registration messages to place a successful registration.
  - This is due to the high waiting time in closest distance buckets and high competition to place ads in these nodes.

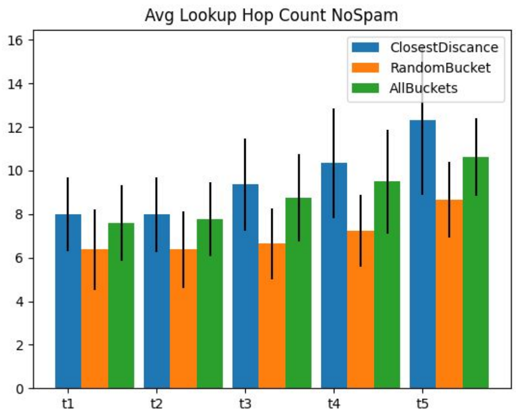


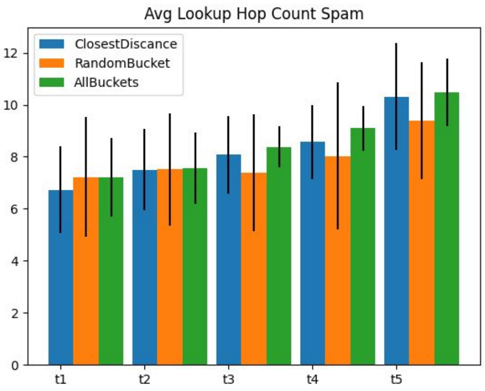


- What we show:
  - The number of messages in the network per type
  - We compare two approaches: Removing Ticket after Registration (Spam) or Removing Ticket after lifetime (No Spam)

- What we observe:
  - We observe there is a lot of overhead generated by repeating ticket registrations.
  - However this is bounded at 20 and 50 messages received per node.
  - Spam approach generates more messages.

**Goal 6**:  the lookup operation should be efficient in terms of time (fast) and messages sent (hop count) for all the query nodes.

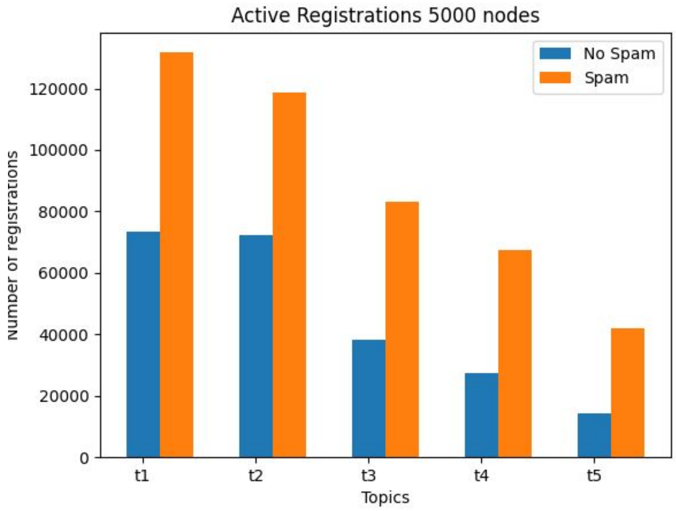




- What we show:
  - Lookup hop count for three different strategies: Closest distance bucket, random bucket and all buckets.
  - We compare two approaches: Removing Ticket after Registration (Spam) or Removing Ticket after lifetime (No Spam)

- What we observe:
  - We observe for No Spam approach, closest distance is faster, especially for non popular topics.

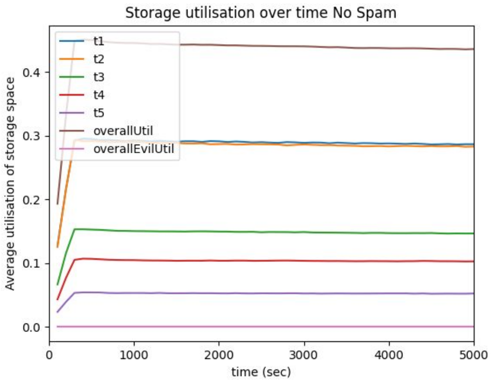
**Goal 7**: the number of total registrations per topic should be proportional to the popularity of the topic (the number of registrants).

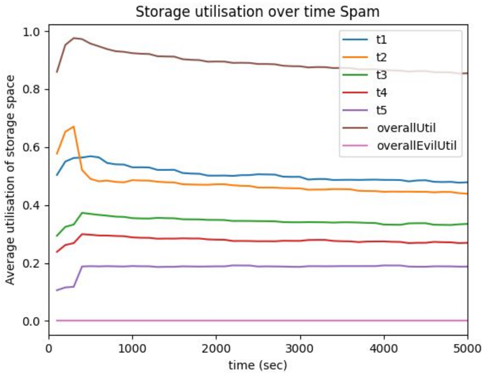


- What we show:
  - Active registrations per topic during simulation time.
  - We compare two approaches: Removing Ticket after Registration (Spam) or Removing Ticket after lifetime (No Spam)

- What we observe:
  - We see topics active registrations are proportional to the popularity of the topic.
  - Spam approach has more active registrations.

**Goal 8**: The protocol should be resistant to network dynamics (nodes joining/leaving)





- What we show:
    - Storage utilisation (topic table occupancy) during time per topic and the overall table utilisation.
    - We compare two approaches: Removing Ticket after Registration (Spam) or Removing Ticket after lifetime (No Spam)

- What we observe:
    - We observe both approaches remain almost constant during time, which means that high amount of turbulence (a turbulence event happens every 1.5 sec) does not affect the performance of the solution and new nodes are able to place registrations and receive registrations.
    - Spam approach utilisation is much higher because more nodes receive registrations (specially for popular topics) meanwhile for no spam approach there are a high number of nodes receiving just a few registrations as shown in Goal 1 graphs.

**Goal 9:** The protocol should be resistant to attacks launched by malicious nodes.

Additional default parameter for scenarios with attackers (in addition to the ones on page 1):

- ID distribution of malicious nodes: uniform
- 20% of nodes are malicious.
- Only Topic 1 (Topics 1 and 2 are most popular) is attacked by malicious nodes.
- No-spam registration strategy.

**Threat Model**: Malicious nodes perform the below activities:

- Malicious nodes are Sybil nodes that cooperate in order to eclipse other “good” nodes.
- Malicious nodes and good nodes have the same amount of bandwidth resources.
- Malicious nodes respond to search (i.e., lookup) requests with only other malicious nodes.
- Malicious nodes respond to Kademlia Find (to build routing table) requests with only malicious neighbors.
- Malicious nodes accept registrations but ignore registered nodes when responding to search queries.

Below, we explore a set of parameters and their impact on the attack power of malicious nodes, where attack power is quantified in terms of percentage of eclipsed nodes out of all nodes in the network.

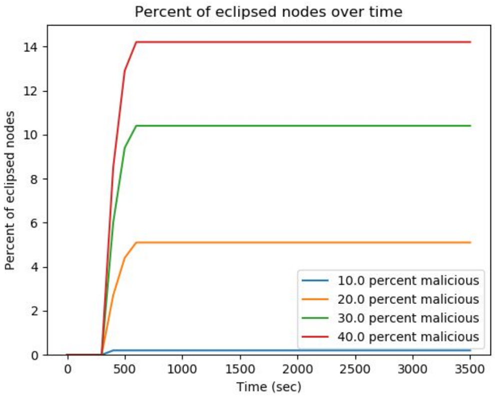
### 1) Impact of ID distribution of Malicious nodes:

We explore two options for ID distribution of malicious nodes: i) IDs of malicious nodes are selected uniformly in the hash space (same as good nodes), and ii) IDs are selected close to the hash of the topic that is being attacked.

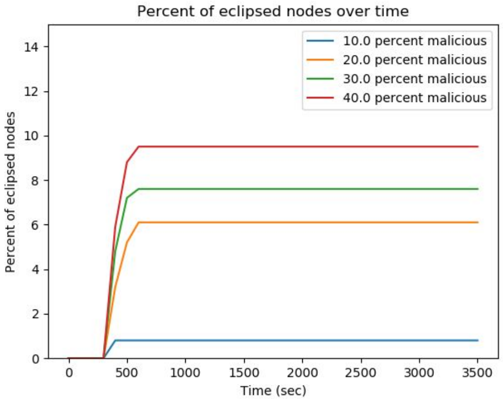
- What we show:
  - ○ The percentage of eclipsed nodes over time with two different ID distribution of malicious nodes and different percentages of malicious nodes (10-40%) .

- What we observe:
  - ○ For the default search strategy (random bucket), we observe that distributing malicious nodes uniformly at random is more effective. This is expected since random lookups do not find many nodes close to the topic hash.

As can be seen below, malicious nodes whose IDs are uniformly (at random) picked from the DHT hash space can eclipse more nodes than the malicious nodes whose IDs are picked non-uniformly (close to the topic hash). In this scenario, all malicious nodes attack the same (most popular) topic.



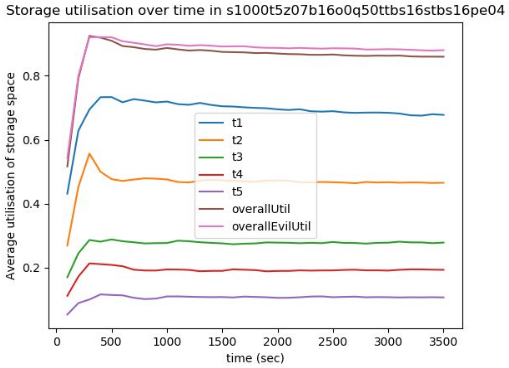
a) Attackers with uniform IDs



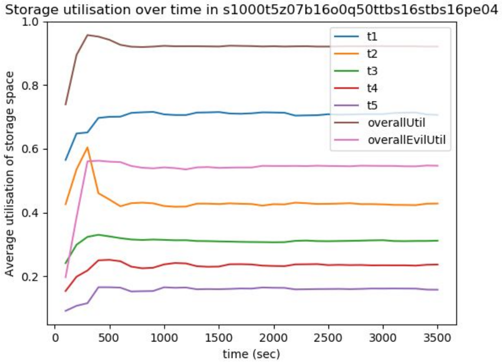
b) Attackers with non-uniform IDs.

- What we show:
    - The storage utilisation of the topic tables by different topics, for good nodes (overallUtil) and evil nodes (overallEvilUtil).

- What we observe:
    - For the default search strategy (random bucket), we observe that good nodes are not able to discover malicious nodes with non-uniform IDs, as can be seen by their storage utilisation (plot on the right below).



a) Attackers with uniform IDs



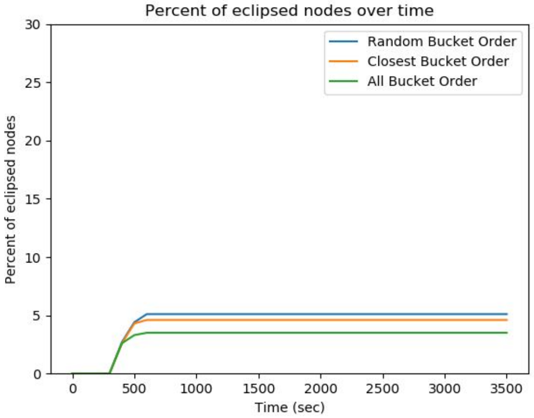
b) Attackers with non-uniform IDs

## Impact of lookup bucket ordering

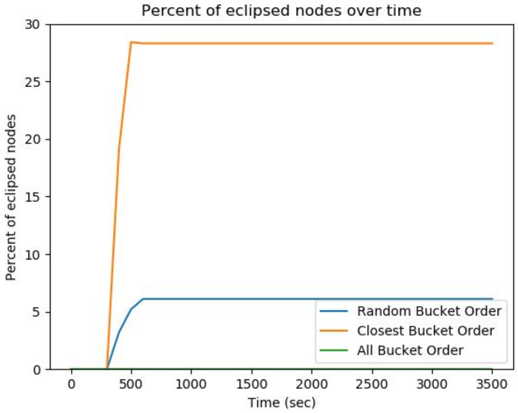
- What we show:
    - The number of eclipsed nodes for different search strategies and ID distributions.

- The number of active registrations (by good and malicious nodes) for different search strategies.

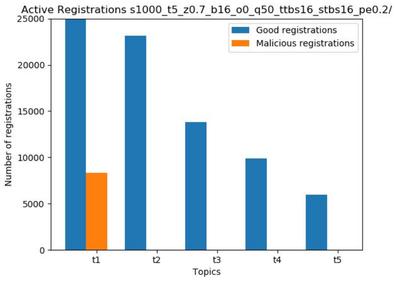
- What we observe:
  - For the scenario with malicious nodes with uniform IDs: all strategies perform similarly and eclipse similar percentages of nodes.
  - In case of malicious nodes with non-uniform IDs: closest bucket order amplifies the attack power of malicious nodes.
  - As can be seen in the three “Active Registration” plots, in each strategy, the number of active registrations by good and malicious nodes are the same for each topic when IDs of malicious nodes are non-uniform.



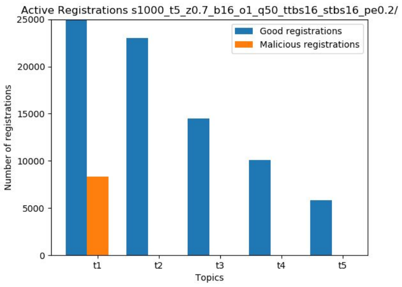
a) Attackers with uniform IDs



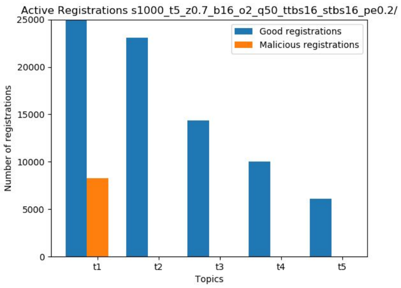
b) Attackers with non-uniform IDs



a) Random Bucket



b) Closest Bucket

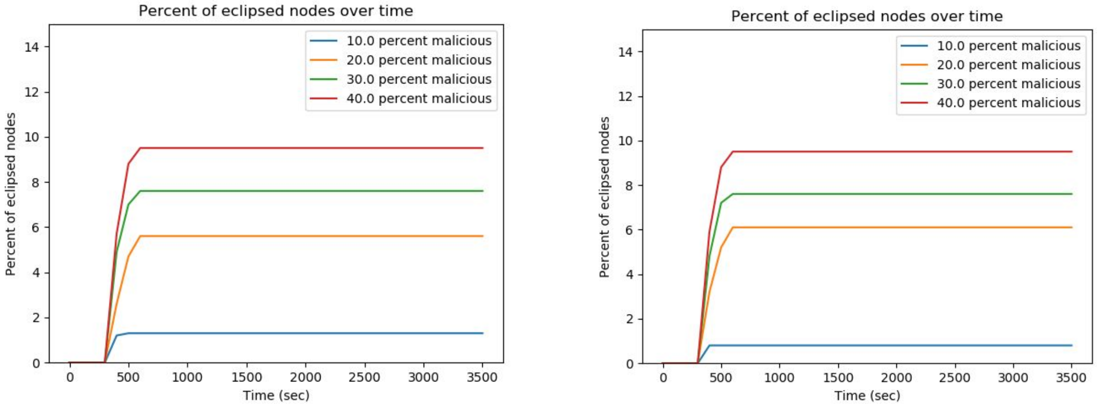


c) All Bucket

**Impact of Registration Method:**

- What we show:
  - The percent of eclipsed nodes for different registration methods (spam and no-spam) and different percentages of malicious nodes (10-40%).
- What we observe:

- For both methods, the number of eclipsed nodes for  the ticketing mechanisms are the same. This is expected because the ticketing mechanism auto-regulates the usage of (topic table) storage resources and provides a fair share of the storage among nodes when the rate of registrations across nodes are uniform.



a) Spamming

b) Normal