P2P - Theorie Aufgabe 6

Arne Link (1582381), Lars Fritche (1691285)
Gruppe 14 (S217/103)

February 4, 2014

1 Aufgabe 6.1 Freenet

1.1 F2F or DHT?

Depending on the extensions used (Opennet or Darknet), Freenet could be seen as both. Opennet is a DHT in a sense, that any node can connect to any other Opennet node, without the other node having to allow this connection. Darknet however is clearly a F2F network, as only explicitly trusted nodes are allowed to directly connect to each other. Therefore: Freenet is a classical DHT, as well as a F2F network. Which one of those to chose (trade off between anonymity and discoverability in case of few trusted nodes) is up to the user of Freenet.

1.2 Storing data

Data is stored by choosing a name for the data to store, then sending this key to a random connected node, including a time to live or hop count in the message. This message is routed by each node to the one node in its routing table that most closely matches the key used for the file name. If the hop count is exhausted, a all clear message will be sent back to the inserting node the same path the storing message took to the target node.

Upon reaching the initiator of the message, this node sends the actual data to the same node it sent the initial store message to. This data is then saved by the first node and a cache entry in the routing table generated, pointing to the source of the data. The data is then propagated to the next node in the path to the target node, repeating the initial procedure.

When the key of the data is already saved, no all-clear message will be sent, but instead the content of the file will be sent back the path, just as if a retrieve query was sent, leading to the alternative content of the data to be stored along the way (not the data in the initial store request).

For privacy reasons, each node can choose to alter the propagated insert message to claim itself or any other random node as data source.

1.3 Retrieving data

Retrieving data is executed in 2 Steps. First, a query message is generated, containing the hash of the file name / description to look up. This message is then, by each node, routed to the neighbor with the best match for this key. Upon encountering loops, a failure message is sent back to the previous node, which then tries the second closest match for the query key.

When the file source (or any node caching this file) is encountered, the file content is sent back the same path the successful query message took, with the data content and source being cached by each node along the way.

For privacy reasons, each node can choose to alter the propagated insert message to claim itself or any other random node as data source.

1.4 Replication

Replication is done automatically and lazily by Freenet. This means, that data is being stored on each node the data is being routed upon. Example:

Node A searches for data with the hashed description "YCTW4". The successful Path for the query goes from Node A to B, C and finally D as the data source. The content is therefore routed from D to C, C to B and finally from B to node A.

C, B and A will then cache the Data content and associated hash, as well as the data source D. B will however (as an example for a random node) choose, not to set D as the content source, but designate Y as the node the message originated from. Node B will choose to set itself as the data source when forwarding the content to A.

On further request for the same data, all Nodes A, B, C, D will be able to satisfy the query and send the data back, but A and C will have different nodes as the data source designated.

The data is therefore replicated on each retrieve and store request, even leading to the original content being replicated in case a malicious node tries to replicate fake data.

This is done do ensure anonymity because other nodes act as data sources so that it is not clear who initially uploaded the data. Another reason for replication is to spread valid data into the network so that in case of false matches or matches with damaged data it is possible to search for better matches.

2 Aufgabe 6.2 Attacking Freenet

2.1 Swapping

Swapping in the Darknet routing algorithm is used to bring nodes similar Location Identifier ($L(n) = \texttt{Location of } n$ with $L(n) \in [0,1)$) closer together. This aids in finding short paths to the target node by increasing the probability that a certain node knows about other nodes similar to itself. This helps to void "dead ends" while searching for a node with a given key.

Swapping works by every node (A) periodically selecting a swapping candidate (B) and checking if the product of distances to its neighbors multiplied by the swapping partners product of distances to its neighbors (D_1) is greater than this product would be after the swap. (D_2)

If this is the case (product distances would be smaller after the swap), the two nodes swap places. If this is not the case (current positions have a smaller product of distances), the places are swapped in $\frac{D_1}{D_2}$ percent of the time. in order to avoid local minima.

Swapping however does not change any established connections to other nodes but only changes the location identifier of a node.

2.2 Attack

The attack facilitates multiple attributes of Freenet and its routing algorithm, leading to a highly unbalanced network with many nodes storing nothing and some nodes having to store much more than they can handle, leading to loss of data.

Nodes have a fixed set of neighbors. Nodes in Freenet (Darknet to be more precise) have a fixed set of trusted nodes they are connected to. These neighbors never change and no neighbor of a node is itself automatically allowed to connect to other neighbors of this node. In fact, neighbors do not even know each other.

Keys and location are chosen randomly.

Locations and Keys are independent. Nodes have a key which determines if they are responsible for storing content, as well as a location identifier, ranging from zero to (excluding) one. Routing in Freenet is done using the location identifier, whereas content storing and retrieval is based upon the nodes key.

These aspects make the following attack possible: A malicious node A (Attacker) tries to cluster every node in the network around a certain location or a certain number of locations. Assuming node A has neighbors V_0 to V_n, it initiates a swap request with a random neighbor, but instead of using its real neighbor information and real location, it picks the location it wants the network to cluster around (lets say, m). This can be done by using a fake neighbor count (ideally more than Vs neighbor count) and faking the neighbors location to be either very close to the $L(V)$ or close to the maximum distance to $L(A)$. This leads to a forced swap of locations, but instead of assuming Vs location, A continues to use location m and initiates another swap request with another random neighbor.

All in all, this leads to the network being tightly clustered around few locations. The key distribution in the network is however still random, leading to a high discrepancy in utilization between nodes. In effect overloading some nodes while starving others.

2.3 Evaluation

The attack is evaluated using a testbed of 18 Linux machines simulating in total 800 nodes. The peers all run the original Freenet 0.7 code.

All node Keys and locations are chosen randomly, while neighbors are chosen to form a small word network. In essence a network where short paths between certain nodes exists, without having to traverse very many nodes. This is done to resemble the structure found in social networks and other social structures, which is adequate for Freenet, as users normally consciously choose their trusted neighbors.

For the attack, a certain number of random nodes (between 2 and 8) are chosen to be attackers, not adhering to the vanilla Freenet switch protocol anymore but actively propagating malicious locations.

In certain intervals (called iterations), routing performance and content loss is measured.