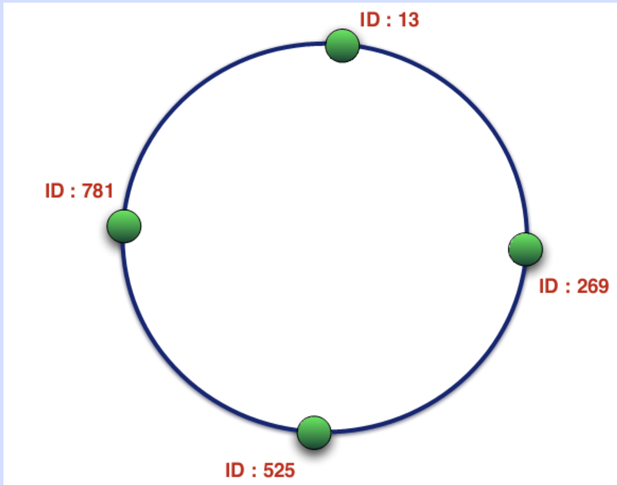
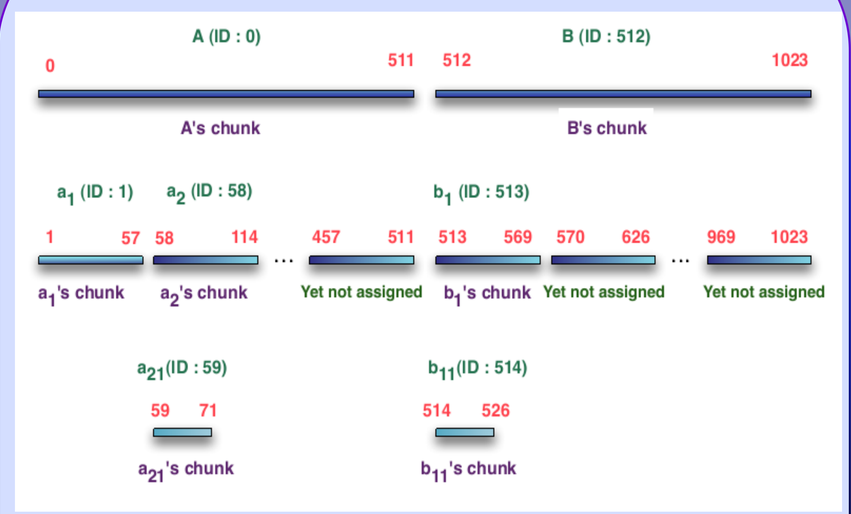
Persea : A Sybil-Resistant Social DHT

:: Background ::

- **Peer-to-peer (P2P) Network :** It provides a lookup service similar to a hash table: (key, value) pairs are stored in a DHT, and any participating node can efficiently retrieve the value associated with a given key.
- **Sybil Attack :** An attacker creates a large number of pseudonymous entities and use them to gain a disproportionately large influence over the system.. By becoming part of the peer-to-peer network, the Sybil attackers can then collude to launch further attacks
- **Attack Edge :** A link between an honest node and a malicious peer. *g* represents total attack edges and *n* stands for total benign nodes.



Replication in evenly-spaced nodes



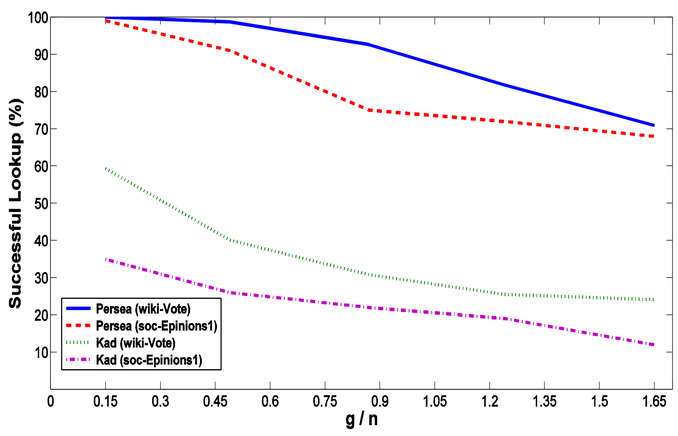
Hierarchical distribution of node IDs through joining-invitation from existing nodes in the network

:: Contributions ::

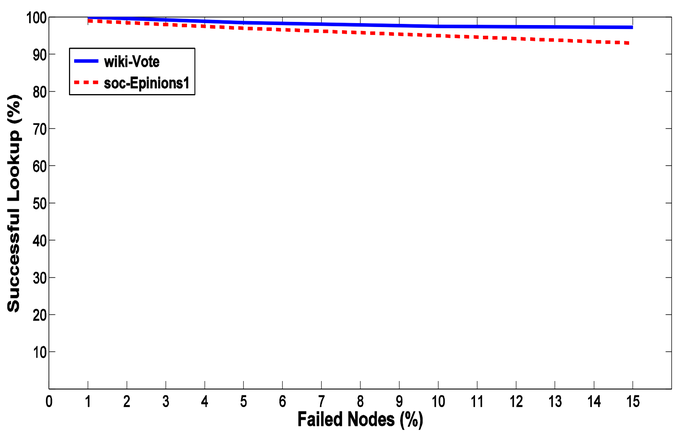
- **Isolated Attacker :** A Sybil attacker is limited to isolated regions of the ID space and must get many invitations to improve his coverage.
- **Fast-Mixing :** Persea does not depend on the assumption that the social networks are fast-mixing.
- **Bootstrap Tree :** Building a bootstrap tree is more realistic than assuming that the clients have access to lists of social network connection from a system like Facebook.
- **Adaptability :** Although we test it with a DHT routing table design similar to Kademlia, which is widely used, it can be adapted to other DHT routing tables.
- **Certified ID :** IDs are certified, making attacks based on ID forging impossible outside of attacker-controlled ID ranges.

:: System Design ::

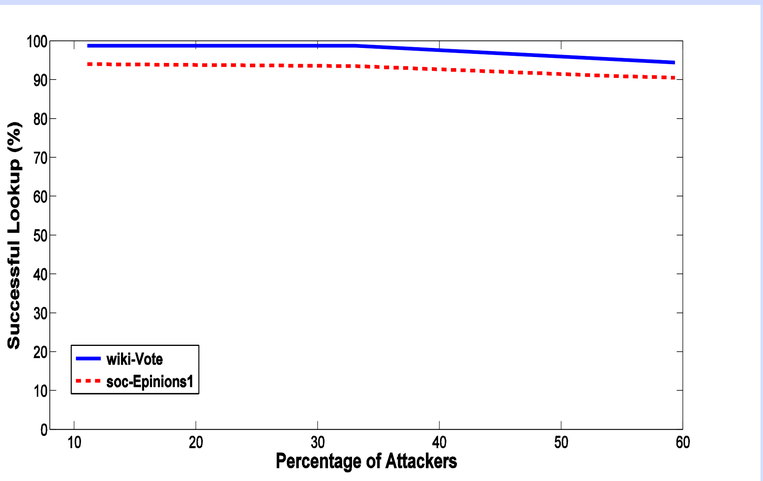
- Persea consists of two layers : the social network layer and the DHT layer. Both layers are built simultaneously starting with a set of bootstrap nodes.
- **Hierarchical Distribution of Node IDs ::** A node joins Persea when it gets an invitation from an existing node in the system. The inviting node assigns a node ID to the joining node and gives it a chunk of node IDs for further distribution. For each chunk of ID space, the attacker needs to socially engineer a connection to another node already in the system. This hierarchical distribution of node IDs confines a large attacker botnet to a considerably smaller region of the ID space than in a normal P2P system.
- **Replication Mechanism ::** The Persea DHT uses a replication mechanism in which each (key, value) pair is stored in nodes that are evenly spaced over the network. Thus, even if a given region is occupied by attackers, the desired (key, value) pair can be retrieved from other regions.
- **Certification of IDs and Chunk Allocation ::** We employ a public key infrastructure. Each node has a certificate, signed by its parent in the bootstrap tree, containing its ID, its public key, the parent's ID, the last ID of its chunk, and a timestamp. The information in the certificate helps to prevent attack based on fraudulent node creation.



Comparison between Persea and Kad



Robustness of Persea against varying fraction of failed nodes



Robustness of Persea against varying attackers for fixed number of attack edges

:: Comparison with Whanau & X-Vine ::

- For g/n = 0.15, lookup Success-rate in Persea is 100%, which is higher than Whanau and X-Vine. Also, for higher value of g/n, lookup success rate in Persea is better.
- In X-Vine, average hop-count per lookup is 10-15, which is 3.84 in Persea..

:: Future Work ::

- Implementing our protocol in a larger network.
- Implementing Eclipse Attack.
- Detail analysis of performance for varying system parameters.