Probabilistic Reputation for Personal Trust Networks

Avani Wildani*(avani@soe.ucsc.edu), Ethan Miller†(elm@soe.ucsc.edu)

January 29, 2009

An issue at the forefront of peer-to-peer storage is trusting the intent and reliability of the nodes you must communicate with. Schemes such as OpenPGP's web of trust lead to flexible, scalable trust, but require out-of-band information that does not readily translate to a peer-to-peer setting. We propose a similar system where individual peers replace this out-of-band communication with a dynamically learned probability distribution of trust values over nodes. We are implementing a decentralized file exchange protocol to demonstrate the power of personalized, learned trust. The key ideas in our system are that nodes make independent decisions about which nodes to trust and those trust tables are kept private and current. This gives nodes the ability to interact with different classes of peers differently based on trust, leading to more robust and decentralized peer-to-peer protocols. If a node is compromised, the most an attacker can learn is that that node trusted a certain group of other nodes. While the attacker can temporarily masquerade as an uncompromised node, the compromised node contains no detailed information about how the trust was obtained and the level of the mutual relationships between the compromised node and its trusted peers. A secondary motivation is to enable peers to use node profiles to detect and avoid nodes used for censorship and information poisoning, maximizing information survival.

The protocol we are using resembles BitTorrent, but it is completely decentralized and focuses on reliability and availability. By having individual peers calculate and store their own reputation data, our system is less susceptible to attacks that could poison a central reputation source. Also, individual learned reputation allows us to treat reputation as intransitive, letting us handle situations where a peer is a member of two disjoint organizations that have no trust relationship with each other. One important future direction for this system is to ensure that those in remote regions who are most vulnerable to information throttling, both incoming and outgoing, are able to fully participate. Thus, our protocols must scale over wide geographic areas and consequently be resistant to slower network links and interrupted connections. Current methods for BitTorrent scaling, such as utilizing all available upload bandwidth and trying to avoid having a peer send duplicate files should be readily adaptable to our system once we alter the fairness mechanisms to represent reputation. A key task will be for the underlying machine learning algorithm to differentiate between unreliable connections and malicious peers.

The first step for a peer ρ to join the system is bootstrapping. This works in one of two ways: either ρ establishes a contact list of peers with varying levels of trust through out-of-band communication similar to friend to friend protocols or ρ broadcasts itself on the system and uses the set of peers with the shortest response time to seed its initial inner circle of slightly trusted peers. When ρ adds data to the system, the data is split into torrent-type chunks, replicated based on the overall system trust level the peer holds, and then sent out to a quasi-random selection of all peers. There will be a slight bias towards peers with high trust that are not holding map data. However, we want to avoid all of copies of a file ending up on a highly connected subnetwork in the trust graph, as this could imply that the subnetwork is controlled by a single organization. The similarly replicated map files for the torrents are sent to the inner circle nodes. If they feel the overall system trust is low, the inner nodes will send a copy of the map to one of their trusted nodes to further protect against information loss. We also intend to experiment with erasure coding to reduce the storage overhead of replicating data. When ρ wants to read a file, it queries its trusted peers for map files, going out progressively to less trusted nodes if the map files are not found. Trust levels are updated on successful reads.

The current implementation focuses on a set of negative indicators a node may use to edit its reputation. The next step is to create a broader node profile to match against to complement known areas of misbehavior. We will then use PlanetLab to simulate a large-scale deployment. PlanetLab will allow us to corrupt nodes with respect to either the entire network or a selected set of peers, which should give us a accurate model from which to reason about the security and feasibility of this protocol.

*Student, University of California, Santa Cruz
†University of California, Santa Cruz