Crawling the Bitcoin Network

Hannah Atmer
ha237@cornell.edu

May 22, 2014

Abstract

The Bitcoin network is a decentralized peer-to-peer network consisting of clients that collectively maintain the state of the network. This paper assesses the propagation and management of addresses in the Bitcoin network. Clients request and broadcast information about transactions, the state of the block chain, peer addresses, and local client state. This paper assesses the mechanisms by which addresses are spread through the network and whether misuse of this feature presents a vulnerability for the network.

1 Bitcoin Protocol

The Bitcoin protocol is run over TCP and Bitcoin protocol packets all use a header that contains a four-byte magic number, the message type, the length of the payload, and a checksum. The protocol messages relevant to this paper are the Version, Verack, GetAddr and Addr messages. Version messages communicate information about the peer it is sent from. Veracks are sent in response to Version requests if the peer that received the Version request is running a compatible client. GetAddr messages are a request for addresses and Addr messages contain a list of addresses.

2 Method

The crawler implemented for this paper mimics the new connection handshake performed by all clients when they join the Bitcoin network. Bitcoin clients are shipped with a hardcoded list of seed addresses that is loaded at initialization. Clients connect to peers at these addresses and send Version packets in order to initiate communication. Peers will either respond with their own Version packets and a Verack packet to acknowledge the connecting peer or terminate the connection if the peers have incompatible versions. Once Version packets have been exchanged they maintain connectivity using ping/pong requests and send requests for information, responses to requests from peers, and gratuitous broadcasts of local information.
Twenty-seven crawlers were deployed and the set of addresses to contact were dynamically partitioned between the crawlers in order to minimize the load on any individual crawler. Crawlers initialize using a list of seed addresses and then create asychronous client connections to each address and perform a Version handshake. Once the handshake is complete the client sends a GetAddr request and then collects the Addr responses received. After 8 seconds the connection is terminated. New addresses are extracted from responses and used for the next iteration.

3 Bitcoin Security

The primary security concern for the Bitcoin network is ensuring that transactions cannot be falsified. The Bitcoin network protects the block chain from tampering by making participating nodes supply a easy to verify but hard to compute proof-of-work such that attacker cannot falsify the block chain unless he controls more than 50% of the computing power in the network. The monetary incentives provided by the Bitcoin network scale according to how much computing power is contributed, so such an attacker would find it more profitable to conform than to defect. While the block chain is quite secure, peer addresses are managed by a probabilistic algorithm that does not provide computing-power based security guarantees.

4 Address Management

Bitcoin clients use a stochastic address manager that limits the number of addresses stored to limit saved state and ensure that a localized attacker cannot fill the address table with his addresses. The address manager stores newly seen addresses in 256 ”new” buckets with up to 64 addresses each, and after an address is verified reachable it swapped with an address from one of 64 ”tried” buckets.

Bucket selection is based on cryptographic hashing from a randomly generated 256-bit key, and only a specific set of 32 ”new” buckets will be accessed for an address sourced from any given /16 subnet. Any nodes not seen alive for three hours are removed from all buckets.

It is clear that while addresses are managed stochastic and protected from localized attackers, each client's address manager will only store a limited number of frequently changing addresses and appears to be vulnerable to a geographically distributed set of attacking nodes. This paper assesses whether or not a distributed crawler like the one implemented for this paper could advertise a list of attacker-controlled nodes to each client it contacts to make those addresses a disproportionate percentage of the addresses propagated through the network. If a large enough number of reachable Bitcoin clients were run and their addresses spammed to the entire network then their addresses would be likely to a majority of the reachable nodes returned by seed nodes. Given that most of the addresses returned in response to GetAddr requests are unreachable, that the average number of reachable nodes found by the crawler was around 4000, and that the standard configuration for Bitcoin clients specifies a default of 125 connections, it is clear that the necessary number of attacker nodes would be limited. The attacking nodes would be configured with a hardcoded list of addresses such that once a client connected to any of the spammed nodes it would receive only the list of attacker nodes. In theory, if this attack were successful then clients connecting to the network could effectively be isolated from the network by the attacker's nodes which would show up as a majority of reachable nodes in the network. Coordination between the attacking nodes could trick the isolated nodes into accepting an inaccurate view of the block chain and arbitrarily modify messages sent from the isolated node before relaying them to the rest of the network. Clients that are configured to only connect to specific nodes would naturally be protected from this type of attack.

Due to constraints on the working environment and ethical concerns, this paper assesses only the possibility that such an attack might succeed, not the effects of a full implementation.

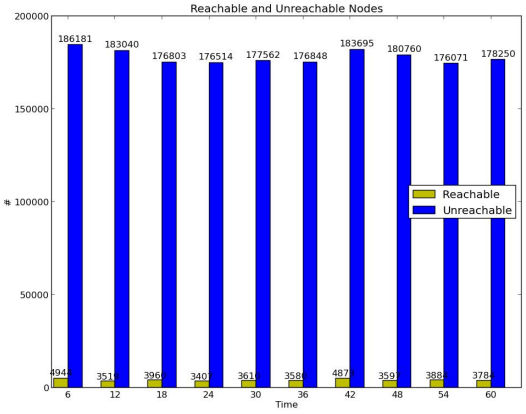
5 Spamming the Bitcoin Network

In order to measure the effects of spamming addresses to crawled nodes, two accessible Bitcoin clients with identical specifications are started. Both clients were configured to accept a maximum of 10000 connections and run on servers with sufficient RAM to handle this number of connections. All crawlers advertised the address of one client to the nodes it contacts after requesting new addresses. The crawlers recorded the number of nodes that advertised the spammed address and the control address in response to GetAddr requests. If the attack described in this paper is possible then the number of nodes advertising the spammed node should be significantly higher than the number of nodes advertising the control address.

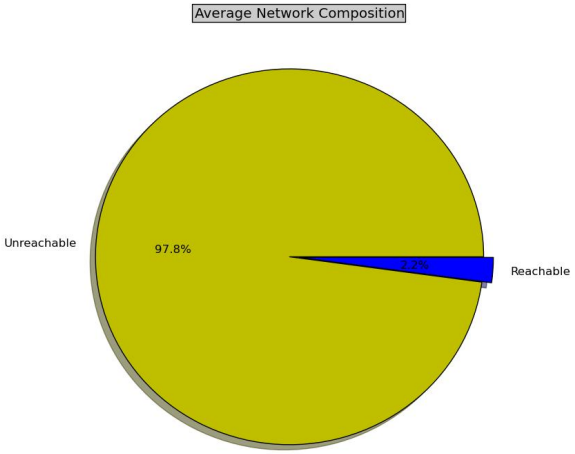
6 Results

6.1 Network Composition

The crawler observed the following statistics over the 60 hour data collection period.



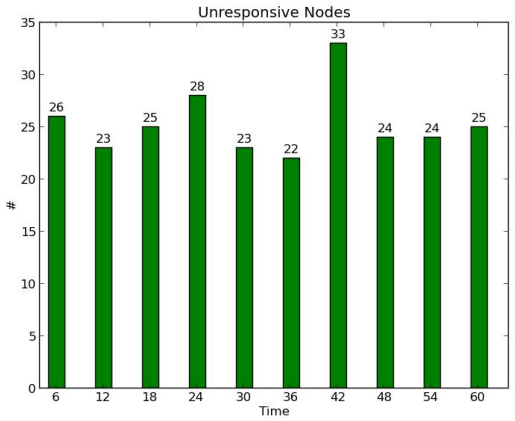
The average number of reachable nodes over the 60 hour data collection period was 3916 and the average number of unreachable nodes was 177786.



6.2 Churn Rates

Taking the intersection of the of reachable nodes and the unreachable nodes for each 6 hour period and for the entire 60 hour data collection period showed that none of the contacted nodes changed from being reachable to unreachable or vice versa.

6.3 Unresponsive Nodes



The set of nodes that accepted crawler connection attempt but did not return any data remained relatively constant over the course of the data collection period. These nodes were likely specially configured clients or passive observers of the Bitcoin network since their addresses continued to appear in advertised address lists. None of the unresponsive nodes were seen in the list of reachable nodes at any time.

6.4 Data Anomalies

The data for subsequent 6-hour data collection periods show an increase in the number of nodes participating in the network at the 42nd hour mark. This timeframe corresponds to the hours between noon and 6pm on May 20th and is likely due to increased rates of users running Bitcoin clients in response to media attention to the Bitcoin currency.

6.5 Spam Effects

Neither the spammed address nor the control address were sent to the crawler by any of the contacted nodes. While the client at the spammed address was observed to fail more rapidly than the control, this was only the case when the maximum number of allowable connections and the maximum buffer size per connection were too high for the servers to maintain without failing. Neither client failed while running with configurations scaled to the server specs.

7 Discussion

The results do not provide support for the hypothesized attack vector presented in this paper. Nonetheless, subsequent trials in which a large number of addresses are spammed instead of just one would likely demonstrate a disproportionate level of visibility for those addresses. This assumption is vaguely supported by the data that shows that contacted nodes did not become unresponsive to the crawler and from the inferred increase in the number of connection attempts to the spammed node. Clearly a single spammed address did not have a noticable effect due to the scale of the network, although the addition of a couple hundred reachable nodes run over a signifcant time period with their addresses spammed to the network using this type of crawler would likely have a noticable impact on the composition of address lists across the network.