How to Pick Your Friends
A Game Theoretic Approach to P2P Overlay Construction

Saar Tochner and **Aviv Zohar**
The Hebrew University of Jerusalem, Israel
{saart, avivz}@cs.huji.ac.il

Abstract

One major limitation of open P2P networks is the lack of strong identities that allows any agent to attack the system by creating many false personas. Such attacks can be used to disrupt the overlay network's connectivity and to sabotage its operation. In this paper, we explore practical ways to defend P2P networks from such attacks. To do so, we employ a game theoretic approach to the management of each peer's list of known nodes and to the overlay construction mechanisms that utilize this list. We consider the interaction between defender and attacker agents as a zero-sum game. We show that the cost of attacks can be driven up substantially if the defender utilizes available information about peers it chooses to connect to such as their IP address. In addition to theoretical analysis of the underlying game, we apply our approach to the Bitcoin P2P network and derive effective strategies that guarantee a high safety level.

1 Introduction

P2P networks are used as an underlying communication layer in many applications such as Bitcoin [Nakamoto, 2008], BitTorrent [Norberg, 2009], and DHTs [Urdaneta *et al.*, 2011]. Unfortunately, they are extremely susceptible to isolation attacks in which individual nodes that wish to participate in the network connect only to attackers and are effectively quarantined from all other honest participants. Attackers can then distort the view of nodes regarding events in the network, filter their messages, change, or delay them at will.

Attacks of this sort give attackers great power. One prominent example, is the Bitcoin protocol [Nakamoto, 2008], in which isolated nodes can have their computational power subverted to attack the rest of the network, or may be blocked from issuing transactions.

The general mechanisms through which P2P clients choose their connections may vary, but most use the following general technique: once an initial connection is established with a node (usually with the help of some centralized server that helps to bootstrap the process) nodes share the IP addresses of others with their connections, and maintain lists of possible peers to connect to. Such potential connections are stored in buffers and are exchanged often to keep the lists populated with "fresh" addresses. The important decisions to be made are how to select which nodes to evict from a full buffer, and which nodes to connect to.

A naive approach is to form connections to random nodes in the buffer (as random graphs are typically well connected) and to evict nodes uniformly at random from the buffer if it is overflowing. Other approaches that have been used are to remove the oldest IPs assuming they are the ones most likely to be stale. Attackers can take advantage of such policies to replace all IPs in the buffer of the victim with IPs of attacker nodes (or with un-assigned IP addresses). The defender will thus fail to connect to other honest nodes and will be quarantined by the attacker.

Bitcoin itself employs a more sophisticated eviction strategy, sorting IP addresses into buckets by combining the IP of the sending node along with the IP address in the message itself. Recent work has shown this mechanism to be susceptible as well due to the fact that attackers get nodes to evict IP addresses from their buffer [Heilman *et al.*, 2015]. There are two approaches used in isolation attacks: Sybil attacks [Douceur, 2002] in which attackers create many identities and thus increase their chance of getting connections and Eclipse attacks [Singh and others, 2006] wherein the attacker increases the visibility of each of its nodes by advertising his own IP more aggressively.

In this paper we seek to make the attacker's job more difficult. We wish to suggest good behaviors for the agents in the P2P network that will successfully avoid being quarantined by attackers, unless the attacker invests a great deal of resources. In particular our work models the limited buffer from which agent we can choose connection and suggests ways to manage this buffer. Our work falls in the general scheme of security games: we consider the attacker and defenders as rational agents that seek to maximize their gains (or alternatively minimize costs and penalties). We assume that if costs are sufficiently high, attackers will be discouraged from attacking or unable to corrupt the required resources.

The main idea that we utilize to increase the cost for the attacker is to take advantage of properties of the connections to peers to try and pick connections that are not just Sybils of the attacker, i.e., to pick a set of nodes that the attacker would be unlikely to control all at once. We take inspiration from Bitcoin's P2P formation, and consider the IP subnet mask of the peers. We assume that once an attacker has purchased an IP address in some range of IPs it is easy for him to gain access to other similar IPs. Our peer selection strategy in this case will tend to be biased towards selecting peers from many different IP ranges. This affects the way we should manage the IP addresses' buffer. Our main result (Theorem 4) states that we can implement a safety level strategy for this game, using only a limited amount of memory, in which the IPs are stored. Our contribution is thus a *practical* one.

It is important to note that our model has been created to defend from attacks on a single node. This is not the general case: the attacker may try to isolate a specific subset of nodes. We thus approximate an honest node as "safe" if it is connected to other honest nodes in the local sense. Our empirical analysis suggests that the resulting network graph is well connected and difficult to attack.

The remainder of the paper is structured as follows: In the next section we briefly review related work, then in section 2 we define our model of the game and the agents' memory buffer. Section 3 shows results for the game without any restrictions on the buffer size. In section 4 we restrict the size of the memory buffer, and derive the safety level strategies for the defender. We conclude and discuss future work in Section 6.

1.1 Related Work

Work by Douceur [2002] was the first to expose the problem of multiple identities (Sybils) in P2P systems with no strong identities. Many such systems have indeed been shown to be vulnerable to such attacks [Castro *et al.*, 2002; Urdaneta *et al.*, 2011].

Bitcoin, a P2P currency system [Nakamoto, 2008] was in fact completely designed to work as an open system with no strong identities. Its overlay formation, however, is still susceptible to attacks [Heilman *et al.*, 2015].

Another approach to defense from Sybils was taken up by 'SybilGuard' [Yu *et al.*, 2006], where peers utilize a social network to form their connections (Sybils are assumed to have few connections to honest participants in this setting). Unfortunately, most settings do not have this additional network of relations between peers to use for overlay formation.

Modern botnets are also known to be structured as P2P networks and their susceptibility to such attacks has been used to attack the botnets themselves [Andriesse and Bos, ; Rossow *et al.*, 2013].

The approach of modeling the interaction between attackers and defenders using game theoretic tools is well established in artificial intelligence [Tambe, 2011]. A line of work on security games deals with several variants of such problems. These include the ARMOR project for security at airports [Pita *et al.*, 2008], transportation networks [Tsai *et al.*, 2009], and Patrolling [Basilico *et al.*, 2009].

2 Model

We assume the attacker wishes to separate the defender node from all other honest nodes. We model this as a game between the defender and the attacker. The first model we examine, assumes (unrealistically) that defender knows all nodes in the "universe" V, and is essentially unrestricted by memory considerations. We later use this as a building block for the second model in which the defender has a limited memory buffer.

2.1 Preliminaries

let S^1, S^2 be the strategy sets of players 1, and 2 correspondingly. Let $\mathcal{U} : S^1 \times S^2 \to \mathbb{R}$ be the utility of player 1 in the game, and assume player 2 has utility $-\mathcal{U}$ (this is a 2-player zero-sum game). Recall that in a 2-player game with strategies S^1, S^2, and utility $\mathcal{U}$ the strategy profile (s^1, s^2) for $s^1 \in S^1$ and $s^2 \in S^2$ is called a *Nash-equilibrium* iff $\forall s \in S^1, \mathcal{U}(s, s^2) \leq \mathcal{U}(s^1, s^2)$ and $\forall s \in S^2, \mathcal{U}(s^1, s) \geq \mathcal{U}(s^1, s^2)$.

Similarly, strategy $s^1 \in S^1$ is called an *L-safety-level strategy* iff $\forall s^2 \in S^2, \mathcal{U}(s^1, s^2) \geq L$ (a similar definition can be stated for player 2).

Recall that in a 2-player zero-sum game the Nash equilibrium (possibly in mixed strategies) is also the max-min solution of the game. This implies that it is comprised of safety level strategies for both players.

2.2 The Limitless-Buffer Game

Denote V as the set of peers (either honest or those owned by the attacker) in the network. Let H be the number of connections the defender creates. We assume the attacker can corrupt or *acquire* a subset S of nodes from the universe V at a cost that we denote by $C(S)$. $C : 2^V \to \mathbb{R}$. We further assume that the attacker gains some value W_{att} from a successful attack against the defender (and that the defender suffers this as a loss).

The 2-player zero-sum game between the attacker and defender is defined as follows: The defender's strategy space S^1 contains possible sets of nodes that it may connect to. $S^1 = \{U \subset V : |U| = H\}$. The attacker's strategy space S^2 contains subsets of the universe V that he chooses to corrupt: $S^2 = 2^V$. The attack is considered successful iff the attacker owns all nodes the defender selected. The utility function is thus

$$\mathcal{U}(s^1, s^2) = \begin{cases} C(s^2) & \text{if } s^1 \not\subset s^2 \\ C(s^2) - W_{att} & \text{if } s^1 \subset s^2 \end{cases}$$

We will usually consider the game with mixed-strategy $\sigma^1 \in \Delta_{S^1}, \sigma^2 \in \Delta_{S^2}$ and assume players maximize their expected utility:

$$\mathcal{U}(\sigma^1, \sigma^2) = \sum_{U \in S^2} \sigma^2_U C(U) - W_{att} \sum_{U \in S^2, V \in S^1} \sigma^1_V \sigma^2_U \delta_{V \subset U}$$

in the above σ^k_U is the probability that player k chooses the subset $U \in S^k$. $\delta_{U \subset V}$ is 1 if $U \subset V$, 0 otherwise.

2.3 The Restricted-Buffer Scenario

We wish to defend P2P networks in realistic settings so we must take buffers and limited knowledge of the world into account. We thus define the following refinement of the game: We assume that at any point in time the defender can only maintain a set of potential peers in his buffer. Let $\mathcal{B}$ denote the buffer size (the number of nodes whose information can be saved). The agent receives a stream of announcements about nodes from which it picks which ones are to be stored in the buffer. We assume honest nodes are advertised at least once in each time period of length $\mathcal{T}$ (attacker nodes may be updated more frequently, as is often the case during eclipse attacks). If a node's details are stored in the buffer, and the buffer is full, a different stored record must be evicted first. The node then chooses its connections from the set of nodes whose records are held in the buffer.

Each node should specify an algorithm $\mathscr{R}$ that implements the functionality that decides, only by the peers' cost and unique identity (the only revealed information), which node records to save or evict from the buffer. As the defender, we try to find $\mathscr{R}$ that maximizes the minimum amount of resources that an attacker should invest to successfully attack.

The cost of corrupting nodes As a main example, we assume that acquiring an IP costs c_{new} and that every further IP from that range is cheaper and costs c_{node}. Mark $D_U := \#$ masks in subset U. Then $C(U) = c_{new} \cdot D_U + c_{node} \times |U|$.

3 Analysis with an Unrestricted Buffer

In this section we show results for the setting in which the nodes in V are known to all.

The following lemma shows that it is better for the defender to err on the side of over-estimating the damage from an attack:

Theorem 1. *Let $\mathcal{U}_{W_i}$ be the utility of the game in which a successful attack causes W_i damage. If (σ^1, σ^2) is a Nash equilibrium in the game with W_1 and $W_2 \leq W_1$, then σ^1 is a $\mathcal{U}_{W_1}(\sigma^1, \sigma^2)$-safety level in $\mathcal{U}_{W_2}$, i.e., $\forall \bar{\sigma}_2 \quad \mathcal{U}_{W_2}(\sigma^1, \bar{\sigma}^2) \geq \mathcal{U}_{W_1}(\sigma^1, \sigma^2)$*

Proof. $\mathcal{U}_{W_2}(\sigma^1, \bar{\sigma}^2) =$

$$\sum_{U \in S^2} \bar{\sigma}^2_U C(U) - W_2 \sum_{U \in S^2, V \in S^1} \sigma^1_V \bar{\sigma}^2_U \delta_{V \subset U} \geq$$

$$\sum_{U \in S^2} \bar{\sigma}^2_i C(U) - W_1 \sum_{U \in S^2, V \in S^1} \sigma^1_V \bar{\sigma}^2_U \delta_{V \subset U} =$$

$$\mathcal{U}_{W_2}(\sigma^1, \bar{\sigma}^2) \geq \mathcal{U}_{W_1}(\sigma^1, \sigma^2)$$

The last step is because (σ_1, σ_2) is an equilibrium. □

Next, we observe that Nash equilibria come in one of two forms: either the attacker corrupts no nodes at all or he places some small probability to "cover" every node.

Lemma 1. *For any Nash equilibrium (σ^1, σ^2) in the game it holds that either $\forall K \quad \sigma^2_K = 0$, or alternatively, $\forall U \in S^1 \ \exists K$ s.t. $U \subseteq K$ and $\sigma^2_K \neq 0$.*

Proof. If exists $U \in S^1$ s.t. for all K with $U \subseteq K$ holds $\sigma^2_K = 0$, then the attacker never answers the defender's strategy to play pure U (denote it with S^1_U). Therefore, $\mathcal{U}(\sigma^1, \sigma^2) \geq \mathcal{U}(S^1_i, \sigma^2) \geq 0$ (the first inequality follows by the Nash equilibrium definition). Clearly $0 = \mathcal{U}(\sigma^1, 0) \geq \mathcal{U}(S^1_i, \sigma^2)$ therefore $\mathcal{U}(\sigma^1, \sigma^2) = 0$. □

We now show that in any Nash equilibrium, the defender places more probability on selecting more expensive sets of nodes (from the attacker's support).

Lemma 2. *Let (σ^1, σ^2) be Nash equilibrium. Then $\forall A, B \in supp(\sigma^2)$:*

$$C(A) \leq C(B) \Longleftrightarrow \sum_{U \subseteq A \cap S^1} \sigma^1_U \leq \sum_{V \subseteq B \cap S^1} \sigma^1_V$$

Proof. Reminder: If v is the value of the game U and (σ^1, σ^2) is a Nash equilibrium, then any pure strategy from the support can be used to achieve it. I.e. if $\sigma^2_A \neq 0$ (for some $A \subset V$), then $\mathcal{U}(\sigma^1, S^2_A) = v$.

$$v = \mathcal{U}(\sigma^1, S^2_A) = C(A) - W_{att} \cdot \sum_{U \subset A \cap S^1} \sigma^1_U$$

$$v = \mathcal{U}(\sigma^1, S^2_B) = C(B) - W_{att} \cdot \sum_{V \subset B \cap S^1} \sigma^1_V$$

The first step in each row is because $A, B \in supp(\sigma^2)$, the second is directly from the definition of $\mathcal{U}$. Therefore

$$C(A) - C(B) = W_{att} \left(\sum_{U \subset A \cap S^1} \sigma^1_U - \sum_{V \subset B \cap S^1} \sigma^1_V \right)$$

So $C(A) \leq C(B) \rightarrow \sum_{U \subset A \cap S^1} \sigma^1_U \leq \sum_{V \subset B \cap S^1} \sigma^1_V$ □

The next corollary is simple, but it contains an insight that we will use later:

Corollary 1. *The probability of choosing a group of peers is determined by the attacker's support, i.e., $\forall (\sigma^1, \sigma^2) \in \Delta_{S^1} \times \Delta_{S^2}$ equilibrium, $\forall A, B \in supp(\sigma^2)$,*

$$C(A) = C(B) \Longleftrightarrow \sum_{U \subseteq A \cap S^1} \sigma^1_U = \sum_{V \subseteq B \cap S^1} \sigma^1_V$$

Reducing the defender's strategy space
In this section, our goal will be to reduce the base of the defender's strategies space so we can decrease the number of nodes we should remember in the buffer, while achieving the same game value.
First, we will define an equivalence relation on S^1 and show that it preserves the game value. Our equivalence relation is defined over sets of nodes $A, B \in S^1$ by $A \sim B \iff \forall(\sigma^1, \sigma^2)$ Nash equilibrium in $\mathcal{U}$, $\sigma^1_A = \sigma^1_B$. We will denote this equivalence class with $[A]$.

Using those equivalence classes as the new defender's strategies space ($\hat{S^1}$) we can define a new game ($\hat{\mathcal{U}}$).

Intuitively, in this game, the defender does not distinguish between different connections' sets in the same equivalence class, so he choses one uniformly. This definition can be also related to the cost to corrupt sets of nodes in any attacker's Nash strategy (as in Corollary 1).

Formally: let $[A]$ be defender's strategy in $\hat{\mathcal{U}}$, U attacker strategy, then define: $\hat{\mathcal{U}}([A],U) = \frac{\sum_{B\in[A]}\mathcal{U}(B,U)}{|[A]|} = C(U) - W_{att}\cdot\frac{\sum_{B\in[A]}\delta_{B\subset U}}{|[A]|}$.

Define a mapping between the defender's strategies in both games: $T:\Delta_{S^1}\to\Delta_{\hat{S^1}}$ by $\left(T(\sigma)\right)_{[A]} = \sum_{b\in[A]}\sigma_B$ and $T^{-1}:\hat{\mathcal{U}}\to\mathcal{U}$ by $\left(T^{-1}(\hat{\sigma})\right)_A = \frac{\hat{\sigma}_{[A]}}{|[A]|}$

The following results show the connection between the two games:

Lemma 3. *$\forall(\hat{\sigma^1},\sigma^2)\in\Delta_{\hat{S^1}}X\Delta_{S^2}$ holds that*

$$\hat{\mathcal{U}}(\hat{\sigma^1},\sigma^2) = \mathcal{U}(T^{-1}(\hat{\sigma^1}),\sigma^2)$$

Proof. First note that:

$$\sum_{U\subset V}\sum_{[A]\in\hat{S^1}}\hat{\sigma^1}_{[A]}\sigma^2_U\frac{\sum_{B\in[A]}\delta_{B\subset U}}{|[A]|} =$$
$$\sum_{U\subset V}\sum_{[A]\in\hat{S^1}}\sum_{B\in[A]}\hat{\sigma^1}_{[A]}\sigma^2_U\frac{\delta_{B\subset U}}{|[A]|} =$$
$$\sum_{U\subset V}\sum_{[A]\in\hat{S^1}}\sum_{B\in[A]}\left(T^{-1}(\hat{\sigma^1})_B\cdot|[A]|\right)\sigma^2_U\frac{\delta_{B\subset U}}{|[A]|} =$$
$$\sum_{U\subset V}\sum_{[A]\in\hat{S^1}}\sum_{B\in[A]}T^{-1}(\hat{\sigma^1})_B\sigma^2_U\delta_{B\subset U} =$$
$$\sum_{U\subset V}\sum_{B\in S^1}T^{-1}(\hat{\sigma^1})_B\sigma^2_U\delta_{B\subset U}$$

Where the second step is $T^{-1}(\hat{\sigma^1})_B = \frac{\hat{\sigma^1}_{[A]}}{|[A]|}$.

Then: $\hat{\mathcal{U}}(\hat{\sigma^1},\sigma^2) =$

$$\sum_{U\subset V}\sigma^2_U C(U) - W_{att}\sum_U\sum_{[A]}\hat{\sigma^1}_{[A]}\sigma^2_U\frac{\sum_{B\in[A]}\delta_{B\subset U}}{|[A]|} =$$
$$\sum_{U\subset V}\sigma^2_U C(U) - W_{att}\sum_U\sum_{B\in S^1}T^{-1}(\hat{\sigma^1})_B\sigma^2_U\delta_{B\subset U} =$$

$\mathcal{U}(T^{-1}(\hat{\sigma^1}),\sigma^2)$. □

The next lemmas are easing the proof of Theorem 2:

Lemma 4. *If (σ^1,σ^2) is Nash equilibrium in $\mathcal{U}$, then $\hat{\mathcal{U}}(T(\sigma^1),\sigma^2) = \mathcal{U}(\sigma^1,\sigma^2)$.*

Proof. Note that $\sigma^1_B = \frac{T(\sigma^1)_{[A]}}{|[A]|}$ for all $B\in[A]$ (because this is a Nash equilibrium), therefore:

$$\sum_{U\subset V}\sum_{B\in S^1}\sigma^1_B\sigma^2_U\delta_{B\subset U} =$$
$$\sum_{[A]\in\hat{S^1}}\Big(\sum_{B\in[A]}\sigma^1_B\big(\sum_{U\subset V}\sigma^2_U\delta_{B\subset U}\big)\Big) =$$
$$\sum_{[A]\in\hat{S^1}}\frac{T(\sigma^1)_{[A]}}{|[A]|}\sum_{B\in[A]}\sum_{U\in V}\sigma^2_U\delta_{B\subset U} =$$
$$\sum_{U\subset V}\sum_{[A]\in\hat{S^1}}T(\sigma^1)_{[A]}\sigma^2_U\frac{\sum_{B\in[A]}\delta_{B\subset U}}{|[A]|}$$

Using this equation in the game utility function as before, and get: $\hat{\mathcal{U}}(T(\sigma^1),\sigma^2) = \mathcal{U}(\sigma^1,\sigma^2)$ □

Lemma 5. *It holds that $\forall\hat{\sigma^1}, T(T^{-1}(\hat{\sigma^1})) = Id$.*

Moreover, if (σ^1,σ^2) Nash equilibrium in $\mathcal{U}$, then $T^{-1}(T(\sigma^1)) = Id$ too.

Clear proof.

Theorem 2. *Nash equilibria in both games have the same game value, and if (σ^1,σ^2) is a Nash equilibrium in $\mathcal{U}$ then $(T(\sigma^1),\sigma^2)$ is a Nash equilibrium in $\hat{\mathcal{U}}$, and holds that $\hat{\mathcal{U}}(T(\sigma^1),\sigma^2) = \mathcal{U}(\sigma^1,\sigma^2)$.*

Proof. At first, we will show that the T function saves the property of Nash equilibrium. Let (σ^1,σ^2) be a Nash equilibrium in $\mathcal{U}$, we want to prove that $(T(\sigma^1),\sigma^2)$ is Nash equilibrium in $\hat{\mathcal{U}}$. On the one hand, $\forall\sigma'^1\in\hat{S^1}$,

$$\hat{\mathcal{U}}(\sigma'^1,\sigma^2) = \mathcal{U}(T^{-1}(\sigma'^1),\sigma^2)\le\mathcal{U}(\sigma^1,\sigma^2) = \hat{\mathcal{U}}(T(\sigma^1),\sigma^2)$$

where the first equality is Lemma 3, the second is the definition of Nash equilibrium in $\mathcal{U}$, and the last is Lemma 4. On the other hand, $\forall\sigma'^2\in S^2$

$$\hat{\mathcal{U}}(T(\sigma^1),\sigma'^2) = \mathcal{U}(\sigma^1,\sigma'^2)\ge\mathcal{U}(\sigma^1,\sigma^2) = \hat{\mathcal{U}}(T(\sigma^1),\sigma^2)$$

where the first equality is Lemmas 5 and 3.

Then we proved that this is a Nash equilibrium. And clearly, they have the same value duo to Lemma 4: $\mathcal{U}(\sigma^1,\sigma^2) = \hat{\mathcal{U}}(T(\sigma^1),\sigma^2)$ □

Corollary 2. *The defender's safety level l in $\hat{\mathcal{U}}$ can be translated to safety level $\ge l$ in $\mathcal{U}$*

Proof. Let $\hat{\sigma}^1\in\hat{S^1}$ be a safety level l in $\hat{\mathcal{U}}$. We should prove that $T^{-1}(\hat{\sigma^1})$ is safety level in $\mathcal{U}$: indeed $\forall\sigma^2\in S^2$, $\mathcal{U}(T^{-1}(\hat{\sigma^1}),\sigma^2) = \hat{\mathcal{U}}(\hat{\sigma^1},\sigma^2)\ge l$ Where the equality on the left is Lemma 3, and the inequality on the right is the definition of safety level l in $\hat{\mathcal{U}}$. □

4 Buffer-Restricted Implementation

We now turn to the scenario where buffers are restricted. We begin by defining new equivalence relation on V: two nodes u,v are equivalent if for any set of other nodes $A\subset V$, adding u to A results in a strategically equivalent set to $A\cup\{v\}$. Formally: $\forall v,u\in V, u\rightleftharpoons v \iff \forall A\subset$

$V, |A| = H-1$ holds $A \cup \{u\} \sim A \cup \{v\}$. We will denote the equivalence class of node $v \in V$ with $[v]_{\rightleftharpoons}$, and the equivalence classes space with $\hat{V} := \{[v]_{\rightleftharpoons} | v \in V\}$.

In general $[\{v_1, \cdots, v_H\}] \neq [v_1]_{\rightleftharpoons} \times \cdots \times [v_H]_{\rightleftharpoons}$ and generally there is no containment in either direction (See lemma 8 for a specific interesting property).

4.1 The Buffer Management Algorithm

In this section we propose a concrete way of implementing the buffer management algorithm $\mathscr{R}$, that utilizes a Bloom filter.

A Bloom filter [Bloom, 1970] is a data structure that uses hash-coded information to encode a set of items. It allows for some small fraction of errors in membership tests (false positives). The error rate can be lowered by increasing memory usage. In our paper, we use this method to avoid nodes that we already saw and accepted including those evicted from the buffer. We use the Bloom filter's deterministic answer to completely avoid known nodes, and accept a small fraction of false-positive answers on new nodes.

First, let us see the main theorem of this section, that discusses the benefits of well-implemented buffer management algorithms. Let $Uni(B)$ denote the uniform distribution on some set B.

Theorem 3. *Assume that we have some buffer management algorithm $\mathscr{R}$ that satisfies the conditions that were presented in subsection 2.3 and in addition $\forall A \in \hat{S}^1$, $Uni([A] \cap \mathscr{R}(v_1, \cdots, v_k)) \sim Uni([A] \cap \{v_1, \cdots, v_k\})$ (i.e. choosing a connection set uniformly in $[A]$ from the set of nodes in the buffer, is the same as choosing uniformly from the entire universe).*

Then we can implement any Nash strategy or safety level on a restricted buffer of size $O(H \cdot |\hat{S}^1|)$ with the same value as the game on limitless buffer.

Proof. Store H nodes for any equivalence class in $\hat{S}^1$. Using the additional given property, $\forall [A] \in \hat{S}^1$ choosing a connection set uniformly in $[A]$ from the set of nodes in the buffer, is the same as choosing uniformly from the entire universe. Therefore, we can play any strategy $\hat{\sigma}^1$ in the game $\hat{\mathcal{U}}$, by uniformly selecting a group in the buffer that is in the chosen equivalence class.

Therefore, we implement a choice that is equivalent to the defender's strategy space $\hat{S}^1$ in the game $\hat{\mathcal{U}}$. Let σ^2 be an attacker strategy, then the value of the game that was played in this buffer-limited world is exactly $\hat{\mathcal{U}}(\hat{\sigma}^1, \sigma^2)$. So finally, if this is a Nash equilibrium strategy, then $\mathcal{U}(T^{-1}(\hat{\sigma}^1), \sigma^2)$ is also Nash equilibrium in $\mathcal{U}$ (Theorem 2). If $\hat{\sigma}^1$ is a $\hat{\mathcal{U}}$ safety level then it is also a $\mathcal{U}$ safety level (Corollary 2). □

We consider the following algorithm: Let EG be the equivalence classes ("buckets") in our game, and let *bucket_size* be the number of nodes that each bucket can hold, and $B \in \mathbb{N}$ the bytes size of the Bloom filter.

```
Initialize:
  BF := Bloom-filter buffer of size $B$.
  $\forall$ bucket $\in$ EG: bucket_history[bucket]=0
  $\forall$ bucket $\in$ EG: buckets[bucket]=$\Phi$
for n := new input node do
  b := n.bucket
  bucket_history[b] ++
  if not BF.contains(n) and Prob($\frac{bucket\_size}{bucket\_history[b]}$)
   then
    if buckets[b].isFull then
      buckets[b].uniformlyRemoveOne
    end
    buckets[b].add(n)
  end
  BF.add(n)
end
```

Algorithm 1: $\mathscr{R}$ algorithm for EG

Lemma 6. *The above algorithm with $EG = \hat{S}^1, B = \infty$[1] satisfies the conditions of theorem 3.*

Proof. We should prove that for any equivalence class in the defender's strategy space ("bucket"), choose uniformly a node from the buffer has an equal probability to choose it uniformly from the entire input. Assume that we make the choice after we saw $v_1, \cdots, v_l$ nodes from this bucket. We need to prove that there is a chance of $\frac{1}{l}$ to choose any node. And indeed, $\forall j \in \{1, \cdots, l\}$, holds that it inserted into the buffer with probability $\frac{H}{j}$ and it still in the buffer after the next input in probability $(1 - \frac{H}{j+1}\frac{1}{H}) = \frac{j}{j+1}$, so after the l'th input: $\frac{j}{j+1}\frac{j+1}{j+2}\cdots\frac{l-1}{l} = \frac{j}{l}$, therefore the total probability of the node v_j to be in the bucket after l inputs is simply $\frac{H}{l}$. Therefore, the probability of choose any v_j uniformly from the buffer is $\frac{1}{l}$, which is exactly the same probability as chose it over all the input $(v_1, \cdots, v_l)$. □

Continuous games: Refreshing the Buffer and the Bloom Filter

The algorithm above works for a single 'round' of choosing the connections, which is not enough for a continuous game, where nodes in the network come and go frequently. To overcome this difficulty, we can define the network protocol to propagate live nodes every $\mathcal{T}$ time units, and store two copies of the buffer and filter that reset alternately every $2\mathcal{T}$ time units. This method gives us the ability to remember IPs from a window of $\mathcal{T}$, which is the full available information on the network (as we've assumed honest nodes send their IP address to others at least once every $\mathcal{T}$).

Additionally, we can use the above algorithm with *buckets_size* $= 1$ because we assume that we need to choose the connection set once, and that all honest nodes will be available to answer. In more realistic scenarios in which churn is an issue, and honest nodes may be offline

[1] a Bloom filter with buffer size ∞ is optimal

at times, we suggest using larger bucket sizes to preserve a sufficiently large set of alternative connections.

In the full version of the paper, we prove that even if some IPs in the bucket can not be selected, e.g. if they are stale, selecting uniformly from the remaining nodes in the bucket is equivalent again to a uniform selection.

4.2 IP masks

For the rest of the paper we focus on the IP mask implementation for the restricted buffer case. A similar treatment applies to other cost functions.

Lemma 7. *If v_1, v_2 are nodes in the same mask then $[v_1]_{\rightleftharpoons} = [v_2]_{\rightleftharpoons}$, i.e., $\forall A \subset V$ with $|A| = H - 1$ and for all (σ^1, σ^2) Nash equilibrium holds $\sigma^1_{A\cup\{v_1\}} = \sigma^1_{A\cup\{v_2\}}$.*

Proof. Assume with contradiction that there exist two nodes $v_1, v_2 \in V$ from the same mask that are not in the same equivalence class. I.e. there exists $A \subset V, |A| = H - 1$ where $A_1 := A \cup \{v_1\}$ and $A_2 := A \cup \{v_2\}$ are not in the same equivalence class. Therefore, exists Nash equilibrium where the defender choose (WLOG) A_1, A_2 in probabilities $\alpha_1 < \alpha_2$. For the attacker, the cost of corrupting groups A_1 and A_2 is identical because v_1 and v_2 are on the same mask. Therefore, an attacker's BR is to play strategy σ^2 where $\sum_{A_1 \subset B} \sigma^2_B > \sum_{A_2 \subset B} \sigma^2_B$ (the cost is identical but the value is higher). Therefore, the defender's best response is to choose A_2 in greater probability than A_1. I.e. the defender's best response is to change his strategy, therefore this is not a Nash equilibrium, and our first assumption is false. □

The following lemma shows that we can save representatives from each equivalence class.

Lemma 8. *For any defender's strategy $A = \{v_1, \cdots, v_H\} \in S^1$, holds that: $[v_1]_{\rightleftharpoons} \times \cdots \times [v_H]_{\rightleftharpoons} \subseteq [A]$ I.e. we can replace any node with a node in the same mask, and still be in the same strategy equivalence class.*

Proof. Our cost function does not distinguish between two nodes in the same mask, we may switch any node with another one in the same mask, and it will cost the same $\implies$ Choosing them unequally will ease the attacker's game (according to Lemma 7) $\implies$ We will choose those two groups with the same probability. □

As a direct consequence of the previous lemma we derive of our main results:

Theorem 4. *We can implement the IP mask game on a limited buffer, with the same game value as the limitless-buffer game.*

Proof. On one hand, we can implement any defender strategy $[v_1, \cdots, v_H] \in \hat{S}^1$ using the classes $[v_1]_{\rightleftharpoons}, \cdots, [v_H]_{\rightleftharpoons}$ (Lemma 8). On the other hand, we can implement any $[v]_{\rightleftharpoons} \in \hat{V}$ using a set of masks (Lemma 7). Therefore, by using the masks as buckets, we can implement any defender strategy in $\hat{S}^1$. □

4.3 Safety level

In this subsection, we define the game $\bar{\mathcal{U}}$ wherein we assume that the defender can choose one node per mask. This restriction on the defender's strategy space gives us a good safety level for the original game, while making computation of the strategy easier.

Note that the defender should not differentiate between two masks that have the same number of nodes (in any Nash equilibrium, masks with the same number of nodes are selected with the same probability). Therefore, the strategic equivalence classes are defined by mask size: the equivalence class of $\{v_1, \cdots v_H\} \subset V$ is all the set $\{u_1, \cdots u_H\}$ where for all i, the nodes v_i, u_i are in masks with the same size.

Therefore, define the equivalence classes game $\bar{\mathcal{U}}$: Allow the defender to choose only one connection in each mask-size. In mask of size a, denote with M_a the number of nodes, and with $avg_a = \frac{c_{new} + a \cdot c_{node}}{a}$ the average cost of node. Then we can bound from below the attacker's cost of corrupting $x_a \in \mathbb{N}$ nodes with $x_a \cdot avg_a$, and the fraction of corrupted nodes with $\frac{x_a}{M_a}$.

We determine the defender's strategy: choose the connections with probability related to the avg values. I.e. select the mask-size a with probability $p_a := \frac{a \cdot M_a \cdot avg_a}{\sum_{i=1}^{\infty} M_i \cdot avg_i}$, and choose a single node uniformly over all the masks.

Denote $y_a \in \{0, 1\}$ as the defender's probability of choosing from mask-size a or not, and x'_a as x_a if $x_a \neq 0$ or $\frac{M_a(1-\mathbb{E}(y_a))}{\mathbb{E}(y_a)}$ otherwise. The game utility is: [2]

$$\bar{\mathcal{U}}(\bar{x}, \bar{y}) = \sum_i x_i \cdot avg_i - W_{att} \cdot \prod_a \left(\frac{x'_a}{M_a}\right)^{y_a} =$$

$$\sum_i x_i \cdot avg_i - W_{att} \cdot \prod_a \mathbb{E}(y_a) \left(\frac{x'_a}{M_a}\right)$$

Note that y_a are Bernoulli distributed parameters. Let $I = [i_1, \cdots, i_r]$ to be a list of indexes of mask-sizes, and denote by p_I the probability to choose from mask of size $i_{I[1]}$, then $i_{I[2]}$, and so on. Then $p_I = \prod_{j=1}^{r} \frac{p_{I[j]}}{1 - \sum_{l=1}^{j-1} p_{I[l]}}$

Therefore, we get $y_a \sim B\left(\sum_{\substack{|I|=H \\ \text{s.t. } a \in I}} p_I\right)$, and then:

$$\mathbb{E}(\bar{\mathcal{U}}) = \sum_i x_i \cdot avg_i - W_{att} \prod_a \left(\sum_{\substack{|I|=H \\ \text{s.t. } a \in I}} p_I \frac{x'_a}{M_a} \right)$$

$$= \sum_i x_i \cdot avg_i - W_{att} \prod_a \left(\sum_{\substack{|I|=H \\ \text{s.t. } a \in I}} \frac{p_I}{M_a} \right) \cdot \prod_a x'_a$$

[2]according to our assumption it's true because $y_i \in \{0, 1\}$. Otherwise it's just an upper bound.

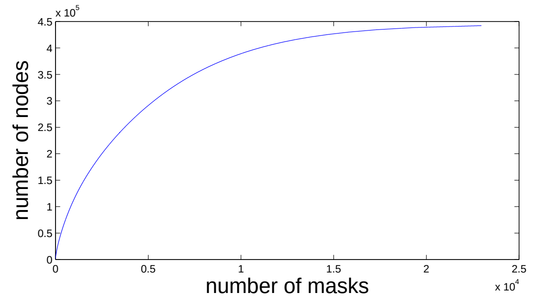

Figure 1: Number of subnet masks\16 in the Bitcoin network (data crawled at September 2015).

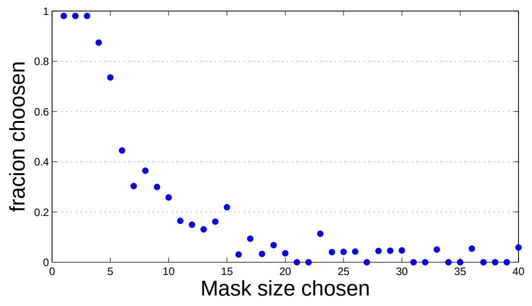

Figure 2: Fraction of times that this mask-size is chosen. Calculated on Bitcoin's network, with $H = 8$, $\frac{c_{new}}{c_{node}} = 10$ (sum of all fraction should be 8).

5 Evaluations

5.1 Evaluation of $\bar{\mathcal{U}}$

To retrieve an actual value for the calculation above, we examine the behavior of the Bitcoin network. We collected a snapshot of all the nodes that connected to the network (including the the IPs appearing on site blockchain.info). From this data we collected statistics on the number of masks, and distribution of nodes within each one (see Figures 1).

We calculated the value of the Bernoulli parameters (Figure 2), and concluded the following term, which is multiplied in the expected utility formula by the profit from the attack: $\prod_a \left(\sum_{\substack{|I|=H \\ \text{s.t. } a \in I}} \frac{p_I}{M_a} \right) \approx 4.357 \cdot e^{-51}$. Therefore only attacks that are highly profitable can derive a negative game value for the defender.

5.2 Evaluating the safety level for Bitcoin

We will use Corollary 2 to deduce the solution for the more general game $\mathcal{U}$. Let $m_1, \cdots, m_k$ be the mask sizes and $\hat{y}_a$ the probability to choose a connection from the a'th mask. Due to the cost function's definition, for all i, j holds $m_i = m_j \Rightarrow \hat{y}_i = \hat{y}_j$, therefore $\hat{y}_a = \frac{y_{m_a}}{|\{i|m_i=S\}|}$.

From here, we derive the probability of choosing IPs from each mask (Figure 3). When we switch back to the original game formulation:

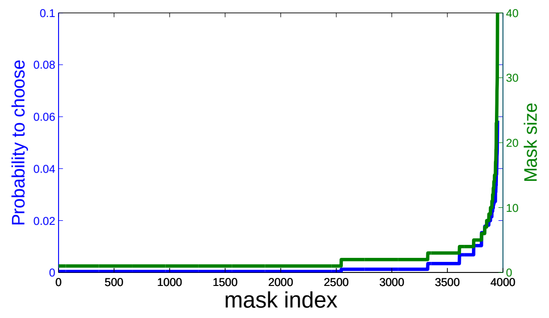

Figure 3: Probability to choose a node from a mask.

$$\mathbb{E}(\mathcal{U}) = \sum_l x_l \cdot avg_l - W_{att} \prod_l \left(\sum_{\substack{|I|=H \\ \text{s.t. } m_l \in I}} \frac{p_I}{M_{m_l}} \right) \prod_l x'_l$$

And finally: $\prod_l \left(\sum_{\substack{|I|=H \\ \text{s.t. } m_l \in I}} \frac{p_I}{M_{m_l}} \right) \approx 1.785 \cdot e^{-8807}$ Therefore the attacker needs to invest great amount of resources to gain a negative game value.

5.3 Comparing to a Naive Benchmark

In this section we would like to compare our results to a naive benchmark, where each node chooses its connections uniformly from the buffer, and the buffer is chosen uniformly from the whole network. In the first benchmark (the most naive), we assume that the node may be subject to repeated transmissions of the same IP, which it can not detect unless that IP is already in the buffer. We assume that the buffer size is the current buffer size of the Bitcoin's nodes (= 20480 unique addresses). For the second benchmark, we assume uniform selection of IPs, but with an accompanying Bloom filter to filter out re-transmissions. We consider a network with the proportion of masks that is derived from Bitcoin's topology. Considering our strategy, an attacker's response to our strategy is to create nodes in a way where p_a is equal for any mask a. Then, we calculate the probability to be successfully attacked as a function of the attacker's investment. The results are shown in figure 4 along with the probability that our own algorithm is successfully attacked. [3]

6 Future Work & Conclusions

In this work we explored a game theoretic model for P2P network formation. Our results indicate that given some model for the cost of nodes for an attacker it is possible to select each peer's connections so as to reduce the likelihood that it is isolated by an attacker.

[3]The 'spike' in the graph caused by the number of masks exists (when the attacker corrupted nodes in all the masks). In this case, the probability to successful attack is the same as in the naive approach.

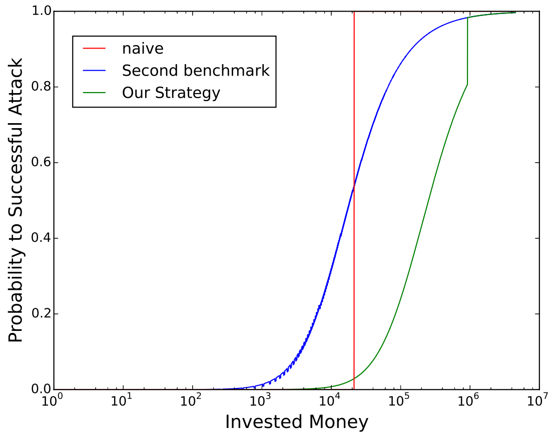

Figure 4: The probability to for a successful attack by money that should be invested ($\frac{c_{new}}{c_{node}} = 10$, $H = 8$).

Future work should extend the model to games that are not zero-sum, and to better account for attackers trying to isolate large chunks of the network (simulations we conducted still show that the strategies that protect single nodes are also good for the network as a whole).

Another possible extension is to consider different information about peers (in addition to IP mask) such as round trip time and other network features, or to augment nodes with additional information that will be hard for an attacker to fake.

References

[Andriesse and Bos,] Dennis Andriesse and Herbert Bos. An analysis of the Zeus peer-to-peer protocol. Technical report.

[Basilico *et al.*, 2009] Nicola Basilico, Nicola Gatti, and Francesco Amigoni. Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 57–64. International Foundation for Autonomous Agents and Multiagent Systems, 2009.

[Bloom, 1970] Burton H Bloom. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, 13(7):422–426, 1970.

[Castro *et al.*, 2002] Miguel Castro, Peter Druschel, Ayalvadi Ganesh, Antony Rowstron, and Dan S Wallach. Secure routing for structured peer-to-peer overlay networks. *ACM SIGOPS Operating Systems Review*, 36(SI):299–314, 2002.

[Douceur, 2002] John R Douceur. The sybil attack. In *Peer-to-peer Systems*, pages 251–260. Springer, 2002.

[Heilman *et al.*, 2015] Ethan Heilman, Alison Kendler, Aviv Zohar, and Sharon Goldberg. Eclipse attacks on bitcoins peer-to-peer network. 2015.

[Nakamoto, 2008] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. *Consulted*, 1(2012):28, 2008.

[Norberg, 2009] Arvid Norberg. utorrent transport protocol. *BitTorrent Extension Protocol*, 29, 2009.

[Pita *et al.*, 2008] James Pita, Manish Jain, Janusz Marecki, Fernando Ordóñez, Christopher Portway, Milind Tambe, Craig Western, Praveen Paruchuri, and Sarit Kraus. Deployed armor protection: the application of a game theoretic model for security at the los angeles international airport. In *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems: industrial track*, pages 125–132. International Foundation for Autonomous Agents and Multiagent Systems, 2008.

[Rossow *et al.*, 2013] Christian Rossow, Dennis Andriesse, Tillmann Werner, Brett Stone-Gross, Daniel Plohmann, Christian J Dietrich, and Herbert Bos. Sok: P2pwned-modeling and evaluating the resilience of peer-to-peer botnets. In *Security and Privacy (SP), 2013 IEEE Symposium on*, pages 97–111. IEEE, 2013.

[Singh and others, 2006] Atul Singh et al. Eclipse attacks on overlay networks: Threats and defenses. In *In IEEE INFOCOM*. Citeseer, 2006.

[Tambe, 2011] Milind Tambe. *Security and game theory: algorithms, deployed systems, lessons learned*. Cambridge University Press, 2011.

[Tsai *et al.*, 2009] Jason Tsai, Shyamsunder Rathi, Christopher Kiekintveld, Fernando Ordonez, and Milind Tambe. Iris - a tool for strategic security allocation in transportation networks. In *The Eighth International Conference on Autonomous Agents and Multiagent Systems - Industry Track*, 2009.

[Urdaneta *et al.*, 2011] Guido Urdaneta, Guillaume Pierre, and Maarten Van Steen. A survey of dht security techniques. *ACM Computing Surveys (CSUR)*, 43(2):8, 2011.

[Yu *et al.*, 2006] Haifeng Yu, Michael Kaminsky, Phillip B Gibbons, and Abraham Flaxman. Sybilguard: defending against sybil attacks via social networks. *ACM SIGCOMM Computer Communication Review*, 36(4):267–278, 2006.