Introduction

- Many applications take advantage of peer-to-peer architectures (e.g., storage, content distribution, live streaming)
- Peer-to-peer design eliminates the need for centralized infrastructures that require expensive servers
- Peer-to-peer organization also introduces the security vulnerability that aggressive resource consumption by a single peer or small group of peers can significantly degrade the performance and availability of the system without being easily detected

Goals

- Limit opportunities for selfish peers to cheat the system by consuming resources without making a proportional contribution in return (i.e., free-riding problem)
- Limit opportunities for malicious peers to disrupt the system by purposely exhausting resources that would otherwise be available to legitimate peers (i.e., DoS attack)
- Accomplish these limitations on selfish and malicious behavior with a scalable, decentralized solution that does not require the use of centralized servers

Framework

- Organize a subset of trusted nodes into another p2p overlay on top of the existing p2p overlay network
- Use these trusted nodes and their corresponding overlay network to collectively detect misbehavior and disseminate information throughout the system regarding attackers
- Once identified, isolate attackers from the system

Protocol

- Each untrusted peer is assigned to be managed by a trusted peer that tracks its upload and download information in a local database
- Prior to granting a request, a peer $P$ will collect a digitally signed ticket $T(R,P,X,A)$ from the requester $R$ indicating that $P$ has provided amount $A$ of some resource or service $X$ to $R$
- $P$ will submit $T$ to its assigned trusted peer $TP$
- $TP$ will update $P$'s record to reflect its contribution
- If $TP$ does not also manage $R$, then it will forward $T$ to trusted peer $TP'$ that manages $R$
- Trusted peer $TP'$ responsible for $R$ will update $R$'s record to reflect its additional resource or service usage
- If $R$’s database record indicates a violation of policy, $TP'$ will alert other nodes

Results

- Simulated 1024-node p2p network with 50 well-behaved nodes making requests and various numbers of attackers also making requests
- Figures show the effectiveness of our protocol at limiting disruptions caused by selfish or malicious peers
- In each figure, $BG$ indicates the percentage of requests granted to well-behaved nodes as we increase the number of attackers while $MG$ indicates the percentage of requests granted to attackers as we increase the number of attackers.

This work was supported by the AT&T Labs Fellowship Program