Security in P2P-Networks

Advanced Computer Networks

Summer Semester 2012

P2P Risks

- Downloading files from and interoperating with other peers bears some risks:
  - Open ports for TCP connection (punched through the firewall)
  - Downloaded files may contain harmful code
  - Downloaded files may contain other content than advertised (typically not that servere)
  - The P2P client may contain malicious code

- Attacker might be member of P2P network!

computer NETWORKS

Malicious Nodes

- Why malicious nodes?
  - „For-fun hackers“
  - Cybercriminals that want to distribute malicious code
  - In file-sharing networks: anti-piracy companies that introduce nodes to infiltrate the network
    - Learn about file-sharers
    - Actively degrade the P2P performance

HowTo Degrade Performance?

- Network Poisoning:
  - Trivial attack
  - Wrong metadata
  - Junk content injection

- Leads to:
  - Resource consumption at client side
  - Quality of P2P network‘s service decreases
    - Users leave the network

Attacker leverages the DHT

- An attacker might leverage the standardized behavior of the DHT
  - Each DHT has routing tables
  - DHTs create an overlay

- Attack:
  - Frequent join/leave
  - Use incorrect routing updates
  - Use incorrect lookup information
  - Drop files that were uploaded for mirroring

„Defense“ solution

- Ensure correctness of routing information
  - „Does that info make sense?“

- Verify existance of nodes before pointing to them

- Verify correct behavior of nodes in periodic intervals (like a ping for correct behavior)

- Against malicious lookup forwarding: incremental search; routing must follow an order that allows to verify that each step approaches the destination

Partitioning attack

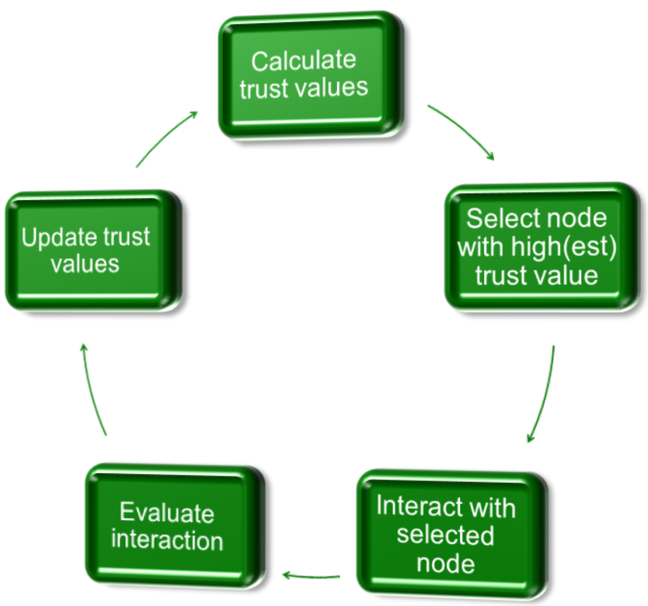
- For initial access, a bootstrapping node needs to be known (or short list of nodes)
- Attack:
  - Take over identity of known bootstrapping node (or become bootstrapping node)
  - Lure every new participant in a parallel, controlled network
  - In the parallel network, information is restricted and service is decreased

Fairness in P2P Networks

- P2P network‘s efficiency is maximized if every node contributes resources
- Problem: Freerider
  - Consumption without contribution
  - (sometime not malicious but due to technical limitations – for example asymmetric bandwidth)
- Requires a solution that provides fairness and load balancing.

Trust and Reputation Systems

- Basic idea of TRS:
  - Observe the long term behavior of nodes
  - Calculate a trust value τ_i that represents the confidence in the future service quality of each node i
  - Preferably interact with nodes of high trust value



Definitions

- Trust definitions (from Jøsang et al., 2007):
  - Reliability: *Trust is the subjective probability by which an individual,* A*, expects that another individual, B, performs a given action on which its welfare depends.* (Gambetta 1988)

    Is high trust sufficient to enter a situation of dependency?

  - Confidence: *Trust is the extent to which one party is willing to depend on something or somebody in a given situation with a feeling of relative security, even though negative consequences are possible*. (derived from McKnight et al. 1996)
- Typically a value [0,1]

Definitions cont'd

- Reputation: An estimation of the collective measure of trustworthiness of a given node
- The reputation value influences the trust value (high reputation typically leads to high trust values)
- To formalize reputation values, the contribution from all P2P members are normalized

A good TRS

- Resnick et al., 2000 identified three properties to operate*:
  - Entities must be long lived.
  - Ratings about current interactions are captured and distributed.
  - Ratings about past interactions must guide decisions about current interactions.

*from: Audun Jøsang, Roslan Ismail, Colin Boyd, A survey of trust and reputation systems for online service provision, Decision Support Systems, Volume 43, Issue 2, March 2007

EigenTrust

- Idea: Calculate a global trust value from the Eigenvalues of the n x n matrix of trust values

- Five considerations for design:
  - Self policing
  - Anonymity
  - No profit for newcomers
  - Minimal overhead
  - Robust to malicious collectives

S. D. Kamvar, M. T. Schlosser, and H. Garcia-Molina, The EigenTrust Algorithm for Reputation Management in P2P Networks, In Proceedings of the Twelfth International World Wide Web Conference, 2003.

EigenTrust cont'd

- Each interaction between peer i and j is evaluated and a satisfaction value is computed:

$$s_{ij} = sat(i,j) - unsat(i,j)$$

- Such local trust values are normalized:

$$c_{ij} = \frac{\max(s_{ij}, 0)}{\sum_j \max(s_{ij}, 0)}$$

EigenTrust cont‘d

- Current state: each peer i has a local values for all nodes that have been in contact with i

- Now: Aggregate local trust values by asking these acquaintances about their opinions and weighting them with their trust estimation:

$$t_{ik} = \sum_j c_{ij} c_{jk}$$

Transitive Trust!

- Please note, that i now can estimate trust of previously not encountered nodes (reputation)

EigenTrust cont‘d

- From the paper:

  We can write this in matrix notation: If we define C to be the matrix $[c_{ij}]$ and $\vec{t_i}$ to be vector containing the values t_{ik}, then $\vec{t_i} = C^T \vec{c_i}$. (Note that $\sum_j t_{ij} = 1$ as desired.)

- That is just a „local view“ but by asking his friend‘s friends and so forth, the vector converges for large n:

$$t = (C^T)^n c_i$$

EigenTrust cont‘d

- For large n, the trust vector converges at every peer to the left principal eigenvector of C

- The trust vector is a global representation of trust for each and every participant!

Basic EigenTrust

$$\vec{t}^{(0)} = \vec{p};$$

repeat

$$\vec{t}^{(k+1)} = C^T \vec{t}^{(k)};$$

$$\vec{t}^{(k+1)} = (1-a)\vec{t}^{(k+1)} + a\vec{p};$$

$$\delta = ||t^{(k+1)} - t^{(k)}||;$$

until $\delta < \epsilon;$

- The p vector provides an a priori notion of trust weighted by a factor a. (helps against malicious collectives)

Distributed EigenTrust

Definitions:

- A_i: set of peers which have downloaded files from peer i
- B_i: set of peers from which peer i has downloaded files

Algorithm:

Each peer i do {

Query all peers $j \in A_i$ for $t_j^{(0)} = p_j$;

repeat

  Compute $t_i^{(k+1)} = (1-a)(c_{1i}t_1^{(k)} + c_{2i}t_2^{(k)} + \ldots + c_{ni}t_n^{(k)}) + ap_i$;

  Send $c_{ij}t_i^{(k+1)}$ to all peers $j \in B_i$;

  Compute $\delta = |t_i^{(k+1)} - t_i^{(k)}|$;

  Wait for all peers $j \in A_i$ to return $c_{ji}t_j^{(k+1)}$;

until $\delta < \epsilon$.;

}

Secured Version

- In the distributed EigenTrust, each peer reports its own trust value – not very secure
- Trust for a node i is maintained by so called score managers that are selected by DHT coordinates (hard to manipulate)
- A network of score managers unfolds as each node has to be handled by multiple managers

Alternative Approaches

- Wide area of research, but most based on concepts similar to EigenTrust
- Sometimes, a notion of Distrust is introduced (values from -1 to 1)
- PowerTrust: uses a trust overlay and leverage the power-law feedback characteristics (some users are significantly more active than others)

Alternatives cont‘d

- PeerTrust: more feedback values than plain „satisfaction“:
  - number of transactions
  - Credibility of feedback
  - transaction context
  - community context factor
  - etc...

Attacks?

- Still possible:
  - Betrayal: Initially behaves well and finally turns malicious (e.g., selling 1000 cheap items at eBay correctly to perform fraud on the following 10 high priced items)
  - Byzantine attacks: cooperation of malicious nodes compromise the system but seem to behave well
  - Whitewashing: After malicious behavior leave the network and return with fresh identity
- Papers typically assume that 2/3 of all nodes are honest

Sybil Attack

- Greatest threat: sybil attack (Douceur 2002)
  - The attacker creates a large amount of fake identities
  - These identities vouch for each other and rate each other well
  - No easy detection of the sybil network
  - Intelligent sybil networks coordinate attacks against individual nodes („destroy reputation“) and behave well in other cases
- High impact on the overall system!

Sybil Defenses

- Still an open research problem!

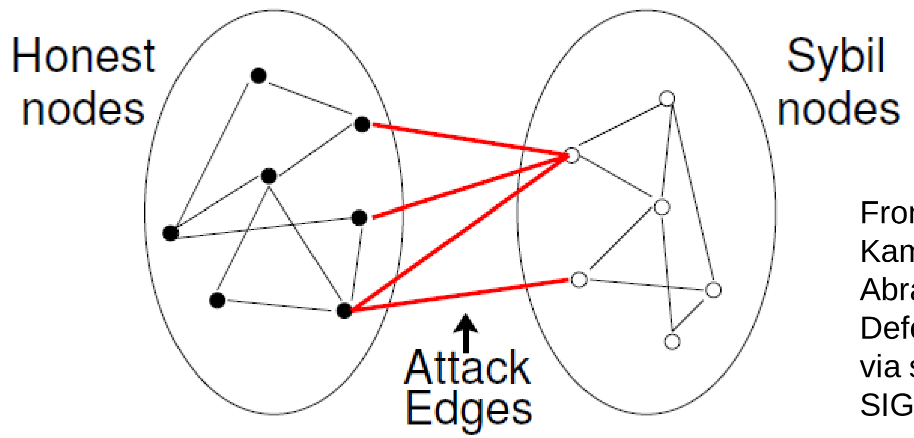
- Two main paths:
  - Introduce a trusted instance that maintains identities and manages the trust values

  - Make it hard to create or operate a large amount of nodes

- Initial step: ensure non-repudiation: each node needs to be held responsible for its behavior and cannot claim to be falsely accused.

SybilGuard

- Use social networks to assist in sybil detection



From: Haifeng Yu , Michael Kaminsky, Phillip B. Gibbons, Abraham Flaxman. Sybilguard: Defending against sybil attacks via social networks, ACM SIGCOMM 2006

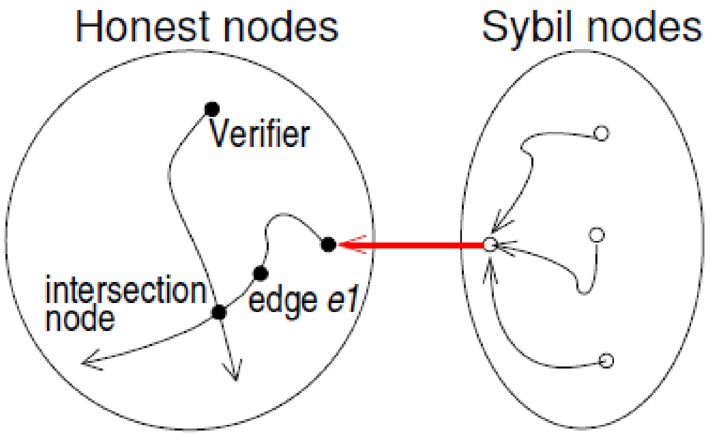
- Somewhat a trick: uses pre-established social trust relationships

SybilGuard cont'd

- Random route: each node has randomized routing table to choose next hop. If a random route comes from the *i*th edge, the edge x_i is used as next hop
- Routing table is generated by fixed permutations
  - Convergence property (two routes from the same input go to the same output)
  - Back-traceability
- Each node is performing random routes of length *w* starting from itself

SybilGuard cont'd

- Another node is accepted, if random routes intersect
  - Therefore: To intersect with the verifies node, the attacker's route has to enter the victim's social network via an attack edge
  - Using multiple random routes allows to exclude the node of which all routes come through a specific edge



SybilGuard cont'd

- Very successful paper
  - Leverages social properties to optimize distributed computing
  - Effectively mitigates the Sybil attack

- But:
  - Uses a pre-established network of trust relationships that needs to be maintained off-line
  - Uses out-of-band distributed symmetric keys for message authentication and non-repudiation

Making Participation Hard

- CAPTCHAs: Human solvable challenges that prevent automatic Sybil generation
  - But: Already observed to be outsourced, for example as a cheap mechanical turk job

- Castro et al. suggest to make identities hard.
  - Public / Private key pair has to comply to the rule that the SHA1 hash of the public key has to end with p zeros at the end.
  - The work required to find such a key is $O(2^p)$

computer NETWORKS

Computational Puzzles

- Problems to consider:
  - Pre-computation
  - Diverse computational capabilities
    - EeePC vs. high speed Desktop
    - Mobile devices…
- Borisov addresses the problem of pre-computation by using all-to-all broadcast of challenges with a combining function that ensures freshness

Excurse

- Aura et al. used computational puzzles in high load scenarios of servers:
  - The puzzles prevent an individual user to DoS the server but allow legitimate users to participate
  - Often considered to be one of the first computational puzzle approaches

Summary

- P2P security is difficult to assure
- There are the classical risks of malicious software, junk content etc.
- There are also attacks on the network infrastructure itself
  - Especially tricky: the Sybil attack
  - Defenses are under research but all current solutions either put a burden of computational puzzles or require third-party knowledge (such as a social network)