

# User Freedom: To Be or Not To Be a 'Supernode'

Khalid Alhamed, Marius C. Silaghi  
Florida Institute of Technology (FIT)

**Abstract**—Peer-to-peer infrastructures in a world where many systems are mobile, or found behind NATs and firewalls that cannot be programmed by users, require significant support from computers with appropriate connectivity, called supernodes. We introduce a new fully decentralized unstructured peer-to-peer (P2P) approach to open-source instant messaging systems (that is employed in the `DirectDemocracyP2P` system). Here each human owning a peer can control the traffic supported by her system. The control may be based on criteria such as: (a) her desire to help the endpoints of the communication based on friendship, (b) her desire to help a cause based on the content/topic of the communication, (c) reputation and rewards, or (d) her interest in the handled data itself. Providing intrinsic incentives for peers to help with traffic is important in order for an open-source freeware P2P system to eventually be viable. In non-incentive P2P systems like Skype, availability of open-source versions can potentially starve the system of supernodes (once users learn how to disable the resource consuming supernode-function). This paper analyzes key functions of the solution such as integration of incentive management into an extension of the STUN protocol, connection establishment and message transfer under different network setups. We show how to use the results of surveys, in conjunction with simulations, to quantify the effects of various incentives on the survivability of an open approach.

## I. INTRODUCTION

Instant messaging is one of the most successful applications on the Internet, starting with `talk` and `IRC` to `Twitter` and `WhatsApp`. Skype is a peer-to-peer (P2P) chat application which has more than 500 million user accounts and has more than 50 million active users per a day [1]. The P2P architecture offers Skype significant robustness and scalability, as well as efficiency, particularly from the perspective of a low latency.

While efforts to build open-source Skype clients have failed due to secret changes in protocols [2], [3], [4], it is questionable whether Skype could in fact have survived its early years as a decentralized service if the protocol would have been open. Namely, with open-source software, the users can easily get versions that disable expensive supernode functions. That can starve the network of supernodes and lead to the demise of the service [1]. Based on our survey of frequent users of Skype, 87% of the users will disable the supernode function as far as they still can have the same benefits.

System functions from which users have incentives to not deviate are said to have the faithfulness property [5]. We introduce a set of *incentives* for peers to run the function of supernode on their personal devices or on dedicated machines. The proposed protocol enables incentives, such as:

- Ability to control the traffic passing by her system:
  - helping the endpoints of the communication (e.g., based on friendship or subject of communication).

- rejecting the endpoints of the communication (e.g., based on disagreement).

- Getting information concerning who talks to whom and what are they talking about (for research, marketing data or pure curiosity) [6], [7].
- Supporting noble causes, e.g. availability of open-source P2P chat systems and the freedom of customization.
- Opportunity to insert advertisements between messages.
- Ability to offer paid services of communication support.

After discussing differences with related work, we detail the proposed incentives management. Further we detail the mechanism used by peers for incentive chat, before presenting experimental results and used metrics.

## II. BACKGROUND

Skype [2] is a P2P application based on the Kazaa architecture for voice over IP (VoIP) and instant messaging (IM). The implementation used by Skype is not open-source and its protocol is not publicly disclosed. In addition, it is not fully decentralized, relying on a set of pre-established nodes for login authentication and as last solution to search registered or logged users [2], [3]. Skype lacks faithfulness [5], in the sense that users benefit by disabling their supernode functions, thereby bringing down the system. A node should have a public IP address in order to become a supernode. Nodes behind NATs or firewalls are reached with versions of the STUN and TURN protocols [8], [9], [10], [11], [4]. If one client is behind a NAT, Skype uses connection reversal whereby the node behind the NAT initiates the TCP/UDP media session regardless of which end requested it.

## III. INCENTIVES MANAGEMENT

The incentives we address are categorized as follows:

- 1) **Curiosity:** e.g. learning who talks what to whom.
- 2) **Commercial perspective:** e.g. opportunity for paid services or inserting advertisements.
- 3) **Altruism:** e.g. supporting others to freely communicate.

Supernodes provide communication services to other nodes based on a set of agreed terms. The parameter *terms* sent with negotiation messages consists of a structure of the type  $OR(term_1, term_2, \dots)$ , i.e., a disjunction of choices, where each  $term_i$  is a pair of the form  $(type, requests)$  where *type* specifies the communication mechanism or information available for the requested peer (direct, forwarded, STUN, or registering), while *requests* is a tuple  $(topic, plaintext, ads, payment)$ , each specifying whether the corresponding incentive is requested and how.

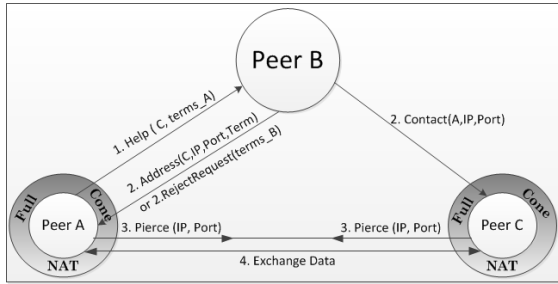


Fig. 2. Reducing latency

#### IV. PROTOCOLS

In this section we introduce the protocol followed by each participant. First we give the perspective of the user. A user follows the sequence of interactions below:

- 1) Acquire **container** of address, e.g, from destination’s website/email. An **address container** is an object (e.g., a file) that contains information about a peer, such as: a list of direct or directory socket addresses.
- 2) For an existing peer, try “direct” connection addresses first, if available. If they work, go to step 5.
- 3) Send **request** message (`Help`) to directories. Directories answer with **negotiation** messages (`Welcome`).
- 4) Choose the directory  $S$  with the best terms and send **agree** message (`Confirmation`).
- 5) Send **data** message based on the obtained procedure.

We illustrate the four cases that can appear in Figure 1 (cases of the STUN protocol, enriched with handling of incentives). These four cases have a simplified handshake mechanism with a higher latency than STUN due to the extra round-trip used to agree on the terms for service.

This handshake can be improved to the latency of STUN whenever the client proactively offers the services that the supernode requests. The example in Figure 2 exemplifies the scenario of full-cone NATs, but it applies identically to the other cases. The `Help` message can optionally contain the list of `terms` that the client (`peer_A`) is ready to accept proactively. Instead of exchanging three messages between peers (`peer_A` and `peer_B`) for terms agreement: `Help`, `Welcome` and `Confirmation`, the client (`peer_A`) only needs to send a single message: `Help` to the supernode (`peer_B`). In case the supernode (`peer_B`) finds acceptable terms among the `terms` offered by the client (`peer_A`), then it returns the most preferred term to `peer_A` using an `Accept` message (in this example, it will return the address of `peer_C`). If the list of `terms` does not match its acceptable terms then it can reply with a `RejectRequest` message specifying its terms. The client (`peer_A`) can eventually retry the communication request including some of the requested terms.

Its user configures `Peer_A` with a list of acceptable terms, tagged with preference values:  $\langle (term_1, p_1), (term_2, p_2), \dots \rangle$ . First the software agent proposes the terms  $term_1$  (with the highest preference  $p_1$ ) to known supernodes serving `peer_C`. If that fails then it tries  $terms_2$  (with the lower preference

	Statistics
Skype users who will disable the supernode function as far as they still can have the same benefits.	87%
Percentage of users who are willing to enable their supernode function based on the causes that they support.	39%
Percentage of users who are willing to enable their supernode function based on helping friends (friendship).	69%
Percentage of users who are willing to enable their supernode function if they get paid for the provided services.	40%
Percentage of users who are willing to enable their supernode function based on revealing the exchanged messages.	29%
Percentage of users who are willing to enable their supernode function for free.	13%
Percentage of users who will not enable their supernode function for any reason.	14%
Percentage of users who are willing to reveal the communication case/topic to get communication services from SNs.	11%
Percentage of users who are seeking help from friends to get communication services from SNs.	55%
Percentage of users who are willing to pay a certain amount of money to get communication services from SNs.	27%
Percentage of users who are willing to receive advertisements from SNs for communication services.	39%
Percentage of users who are willing to send plaintext (unencrypted messages) through SNs to get communication services from these SNs.	9%

TABLE I  
SURVEY STATISTICS (HTTP://DIRECTDEMOCRACYP2P.NET/SURVEY/)

value  $p_2$ ), until it exhausts the acceptable terms. The user is presented with the terms suggested by supernodes in their rejection replies. Users can change their default terms for certain supernodes, based on answers to past requests. The agent can optimize query of supernodes based on past answers.

#### V. METHODOLOGY AND EXPERIMENTS

We now describe the methodology we propose for evaluating the impact of given incentives on the survivability of an open-source system. We exemplify with the case study of our instant messaging system. To quantify the effects of proposed incentives on the survivability of the proposed approach, we have conducted a randomized survey of  $M = 223$  frequent users of Skype. The results of the survey, in conjunction with simulations are used as a base for our evaluations.

##### A. Survey Statistics:

The questions and statistics of our sample are shown in Table I. The popularity of incentives is compared in Figure 3.

##### B. System Simulation:

Due to the intrinsic privacy of the system one cannot evaluate the incentives directly on their implementation in DirectDemocracyP2P system [12]. Assuming users behave as per our survey feedback, we simulate our incentives-based-chat system to evaluate its survivability. The simulation has:

- $N$  peers ( $P_1..P_N$ ), each participant  $i$  out of the  $M$  participants in the survey is replicated by the set of peers  $\{P_j \mid \exists k \in \mathbb{N}, j = i + k * M, j \leq N\}$ .
- A subset of  $S$  instances of these peers voluntarily act as supernodes based on the answers from the survey

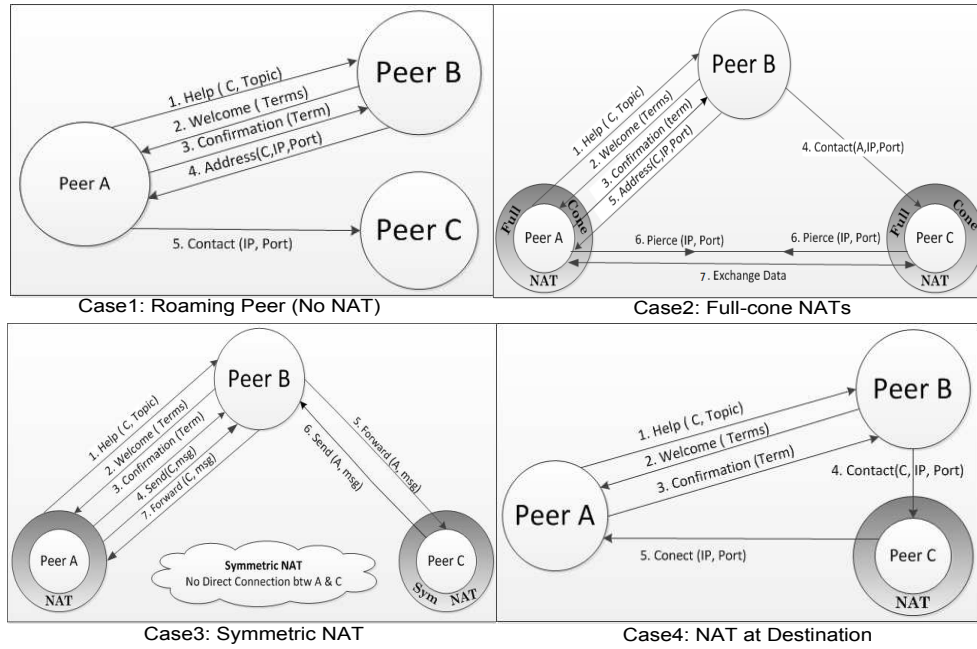


Fig. 1. Connection cases where supernode help is required

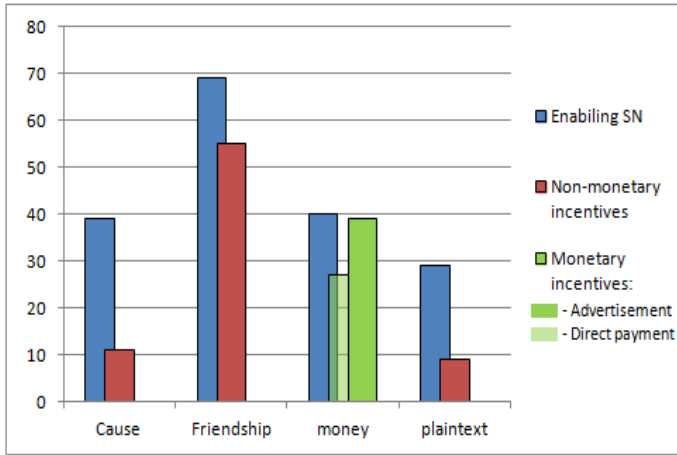


Fig. 3. **Comparing Incentives:** Based on the survey, the chart shows the percentage of the agreement on each incentive from two perspectives: 1) Incentives accepted by prospective supernode owners for enabling the supernode function (blue) and 2) Incentives offered to get the necessary communication services from supernodes, with two types of incentives: a) Monetary Incentives (green), including direct money payment (light green) or payment by advertising (dark green) and b) Non-monetary incentives (brown), including declaring a subject of the communication, exchanging unencrypted messages and getting help based on friendship.

related to: 1) number of instances (devices) in the system. 2) Peer-instance willingness to be a supernode.

- Each peer  $P_i$  is linked to  $n_{(i \bmod M)+1}$  other peers, where  $n_{(i \bmod M)+1}$  is the answer of the survey question regarding the number of contacts the participant has in her messaging system.
- Each peer  $P_i$  has  $m_{(i \bmod M)+1}$  instances, where  $m_{(i \bmod M)+1}$  is the number of devices declared by the

corresponding survey participant.

- Each peer  $P_i$  is registered with  $k_i$  supernodes, based on need and test parameters being evaluated.

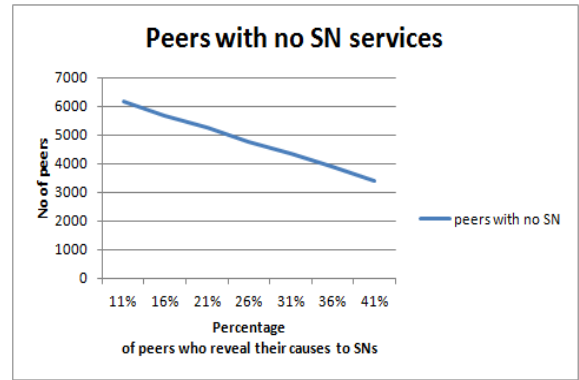


Fig. 4. **Only use "revealing cause" as an incentive:** The chart shows the number of peers who cannot register with any SN based on the percentage of peers who are willing to reveal their causes to supernodes. This experiment is enabling only "revealing cause" as an incentive.

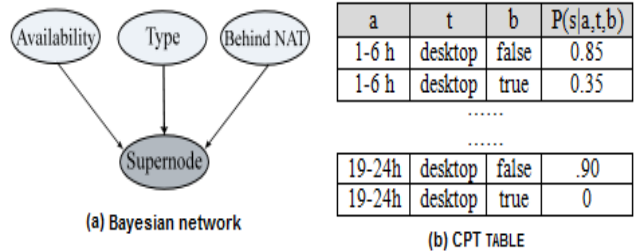


Fig. 5. Bayesian network and conditional probability table (CPT)

		Incentives					
		Using all incentives	Only use monetary incentive	Only use advertisement incentive	Only use friendship incentive	Only use cause incentive	Only use plaintext incentive
Consequences	Avg. number of SNs a peer registers to	3.6712 (SNs/peer)	2.2101 (SNs/peer)	2.599 (SNs/peer)	3.0832 (SNs/peer)	1.6621 (SNs/peer)	1.6034 (SNs/peer)
	Maximum number of SNs a peer registers to	15 (SNs/peer)	15 (SNs/peer)	14 (SNs/peer)	16 (SNs/peer)	14 (SNs/peer)	14 (SNs/peer)
	Avg. coverage time for each peer	24.0 (hours/day)	13.125 (hours/day)	15.876 (hours/day)	19.5341 (hours/day)	9.1728 (hours/day)	8.9952 (hours/day)
	Coverage of least served peer	24 (hours)	18 (hours)	24(hours)	6 (hours)	24 (hours)	24 (hours)
	Percentage of peers covered 24 hours	100 %	54.68%	66.15%	80.74%	38.22%	37.48%
	Number of peers who cannot register with any SN	0	4531	3385	1845	6178	6252

TABLE II  
UNIQUE INCENTIVE

		Using all incentives (excluding free-services)	Absence of incentives				
			Absence of monetary incentive	Absence of advertisement incentive	Absence of friendship incentive	Absence of cause incentive	Absence of plaintext incentive
Consequences	Avg. number of SNs a peer registers to	3.4968 (SNs/peer)	3.046 (SNs/peer)	2.8121 (SNs/peer)	2.4568 (SNs/peer)	3.3803 (SNs/peer)	3.3913 (SNs/peer)
	Maximum number of SNs a peer registers to	13 (SNs/peer)	12 (SNs/peer)	12 (SNs/peer)	13 (SNs/peer)	13 (SNs/peer)	12 (SNs/peer)
	Avg. coverage time for each peer	23.1234 (hours/day)	20.5346 (hours/day)	18.8203 (hours/day)	16.032 (hours/day)	22.3724 (hours/day)	22.4807 (hours/day)
	Coverage of least served peer	12 (hours)	12 (hours)	15(hours)	24 (hours)	15 (hours)	18 (hours)
	Percentage of peers covered 24 hours	95.8%	84.86%	77.83%	66.80%	92.65%	93.19%
	Number of peers who cannot register with any SN	358	1435	2151	3320	671	627

TABLE III  
ABSENCE OF INCENTIVES: NOT ACCOUNTING FOR FREE SERVICES

1) **Supernode Candidacy:** Any regular node with a public IP address having sufficient CPU, memory, uptime and network bandwidth is a candidate to become a supernode [2], [13]. We introduce a quantitative measure of the degree to which a node is desirable to function as a supernode, called: *candidacy level*. For example, a powerful desktop connected to the Internet 12 hours/day has a different supernode candidacy level than a slower device (e.g. common laptop) connected to the Internet 2 hours/day. We summarize three used variables:

- 1) Type (Power): It has 4 possible values: desktop, laptop, tablet and cellphone. Cellphones can be built to have more computation power than some desktops, but we assume that in general these values correspond to a decreasing computational power.
- 2) Availability: the time that a peer is connected to the Internet daily (with assumption of enough network bandwidth). This variable has 4 possible values: 1-6 hours, 7-12 hours, 13-18 hours and 19-24 hours.
- 3) Behind NAT: It is reported [14] that 60% of P2P users are behind some kind of NAT. Can be: true or false.

The simple Bayesian Network in Figure 5 models the probabilistic relationships between the variables: type (t), availability (a) and behind NAT (b), and the candidacy level of a supernode (s). Figure 5 illustrates a few rows in a sample

conditional probability table (CPT) to represent the probability of the candidacy level of a supernode with respect to the other three variables,  $P(s | t, a, b)$ . The table has 32 rows ( $4 \times 4 \times 2$ ).

For example, a desktop computer which has a public IP (not behind a NAT or firewall), running and connecting to the Internet for 5 hours, has a high probability to be a reliable supernode. E.g.,  $P(s | \text{"desktop"}, \text{"1-6"}, \text{false}) = 0.85$ . The probability of being a reliable supernode for a desktop available for 5 hours but running behind some kind of NAT or firewall (potentially with port forwarding and firewall exceptions) is smaller, e.g.,  $P(s | \text{"desktop"}, \text{"1-6"}, \text{true}) = 0.35$ . The numerical data is to be extracted from surveys.

2) **Measures:** The measures used to evaluate the impact of each incentive on the survivability of the system are:

- **Number of peers that find no supernode.** This measure shows how many peers find no supernode serving for the incentives that the peers are willing to offer.
- **Coverage of least served peer.** This measure shows the number of hours of coverage for the peer with the smallest number of covered hours.
- **Average percentage of covered time.**
- **Percentage of peers covered 24h.** This measure shows how many peers are covered at any time.
- **Maximum number of needed supernodes.** This measure shows the largest number of supernodes that a peer

had to register with, to achieve its maximum coverage.

- **Average number of needed supernodes.** This measure shows the average number of supernodes that peers had to register with, to achieve their maximum coverage.
- 3) *Experiments:* We evaluate for  $M=223$  and  $N=10000$ :
- **Unique Incentive.** In this experiment only one type of incentive is enabled at a time (besides the supernodes that serve liberally). The results in Table II reveal the importance of supporting friendship and advertisements.
  - **Absence of Incentives.** This experiment measures the impact of the lack of support in the system for a given incentive when all other considered incentives (except for free service) are supported. This experiment can show the overlap between the populations covered by the given incentives, and can help remove redundancy (incentives that make no difference). The results in Table III confirm that friendship and advertisements are powerful mechanisms to match peers to supernodes. People claim to be more inclined to pay money or spend time on advertisements rather than to explicitly lose some of their privacy. When supporting free service, the removal of any single incentive support still allows for everybody to be covered for 24 hours with an average of approximately 4 supernodes, and a maximum of 16 supernodes.
  - **Robustness of Incentive Declaration.** The experiment evaluates the robustness of our simulation results to the accuracy of the declarations by participants in the survey. We add/remove some incentives of the given types from the lists of incentives that participants claimed to be willing to offer, and we check the slope of the curves that this generates for a given measure. For example, in Figure 4 we compute the impact of variations in declarations of pursued *noble causes* on the number of peers that find no supernode to serve them. The found impact is smooth, implying that no big change comes from small errors in the survey. However, the impact for large errors (11% - 41%) changes results by a 2:1 ratio.

## VI. CONCLUSIONS

We address the problem of providing incentives for users to let their systems serve as supernodes (helpers for initiating or forwarding communication) in an open-source P2P protocol. Skype, a popular close source P2P chat application, does not have the *faithfulness* property. Namely, the only reason for which most users in our sample allow their Skype agent to relay messages is that they cannot turn it off.

How to provide intrinsic incentives for peers to serve as supernodes, to make an open-source chat protocol viable? A human benefit for volunteering to send data is the good feeling of *servicing a noble cause*. While some volunteers would simply offer their resources for the cause of *open-source*, others may want to learn a declared *topic* of the discussion, or to even request access to the whole communication (in plaintext), to guarantee that they are happy to support the *cause* of the given connection. This may be acceptable in certain cases. Another potential motivation of a volunteer is *curiosity* of who talks to

whom, of how much they talk and, if public, on what topics. Closed software always can obtain this data, but in our case users control what information they give away.

A third intrinsic benefit of the supernodes can be *commercial*. The collected data can be used for marketing, studies, etc. Supernode users can also get credit for a *quid pro quo* help if in the future they will be traveling or lack a publicly addressable IP. Moreover, the supernodes can be enabled to insert advertisements into the stream of data.

Based on volunteer supernodes one can now establish an incentive fully decentralized open-source system for chat.

A significant contribution of this study consists of a methodology to quantify and compare the power of different incentives. Based on our study and methodology, a designer of a P2P system can assign priorities to the incentives that it wants to support, such that he can efficiently assure the faithfulness and survivability of an open P2P service. The methodology is based on surveys, and simulations integrating the results of these surveys into a model of the system. We also provided details on a case study implementation for a chat protocol exploiting the studied incentives [12]. Our experiments for this case shows that the incentive with the most impact is friendship, followed by the support of benefits via advertisements. Other direct monetary incentives come far behind, trailed by the loss of privacy via revelation of plaintext and causes/topic.

## REFERENCES

- [1] Y.-K. R. Kwok, *Peer-to-Peer Computing: Applications, Architecture, Protocols, and Challenges*. CRC Press, 2012.
- [2] S. A. Baset and H. G. Schulzrinne, "An analysis of the skype peer-to-peer internet telephony protocol," in *TR UCUCS-039-04*, 2004.
- [3] —, "An analysis of the skype peer-to-peer internet telephony protocol," in *INFOCOM*, 2006.
- [4] S. Guha, N. Daswani, and R. Jain, "An experimental study of the skype peer-to-peer voip system," in *IPTPS 2006*, 2006.
- [5] J. Shneidman and D. C. Parkes, "Specification faithfulness in networks with rational nodes," in *Proceedings of the Twenty-third Annual ACM Symposium on Principles of Distributed Computing*. ACM, 2004.
- [6] F. news, "Microsoft says it snooped on hotmail to track leaker," Fox news, 2014. [Online]. Available: <http://www.foxnews.com/tech/2014/03/21/microsoft-says-it-snooped-on-hotmail/>
- [7] D. Rushe, "Skype's secret project chess reportedly helped nsa access customers' data," <http://www.theguardian.com/technology/2013/jun/20/skype-nsa-access-user-data>, June 2013.
- [8] J. Rosenberg, R. Mahy, P. Matthews, and D. Wing, "Session traversal utilities for nat (stun)," IETF RFC 3489, October 2008.
- [9] R. Mahy, P. Matthews, and J. Rosenberg, "Traversal using relays around nat (turn)," IETF, RFC 5766, 2010.
- [10] J. Rosenberg, "Interactive connectivity establishment (ICE): a methodology for network address translation (NAT) traversal for the session initiation protocol (SIP)," IETF, 2003.
- [11] B. Ford, P. Srisuresh, and D. Kegel, "Peer-to-peer communication across network address translators," in *Proceedings of the Annual Conference on USENIX Annual Technical Conference*. USENIX Association, 2005.
- [12] M. Silaghi, K. Alhamed, O. Dhannoon, S. Qin, R. Vishen, R. Knowles, I. Hussien, Y. Yang, T. Matsui, M. Yokoo, and K. Hirayama, "Direct-DemocracyP2P, decentralized deliberative petition drives," in *P2P*, 2013.
- [13] www.skype.com, "It administratorsguide," Skype, 2010. [Online]. Available: <http://download.skype.com/share/business/guides/skype-it-administrators-guide.pdf>
- [14] Y. Tang Yun, J.-G. Luo, M. Zhang Meng, M. Zhang, and S.-Q. Yang, "Deploying p2p networks for large-scale live video-streaming service [peer-to-peer multimedia streaming]," *Communications Magazine, IEEE*, vol. 45, no. 6, pp. 100–106, June 2007.