



Diploma Thesis for Daniel Eichhorn

| | |
|-------------------|--|
| Task Description: | Thomas Bocek, Martin Waldburger, Prof. Dr. Burkhard Stiller |
| Topic: | A Peer-to-Peer Network Framework with Network Address Translation Traversal |
| Start Date: | November 15, 2005 |
| End Date: | May 15, 2006 |
| Supervisor: | Thomas Bocek, Martin Waldburger |
| Location: | Institut für Informatik |
| Support: | Distributed hash table, super peers, XML, java.nio, GPL |

1. Introduction and Motivation

In the last few years Peer-to-peer (P2P) networks have earned a bad reputation of being solely used for illegal file sharing activities. However, P2P systems can be used for legal activities and much more than sharing mp3 files. One application area being already used today is distributing huge amounts of data at a very low cost. Some companies like RedHat are using P2P technology to distribute their software [2]. Other application fields, like broadcasting media streams through P2P systems or search engines based on distributed systems, are waiting in the wings.

P2P networks are an emerging technology. Fundamental research has already been made and a basis for further research is present. However, there are still many open research fields in this area, especially on the application side, for optimization, and in the area of security.

The basic of every P2P network is that it consists of equitable peers that are interconnected. In contrast to client-server architecture, every peer in a P2P system can be server *or* client.

There are three types of P2P networks: structured, semi-structured, and unstructured. In the beginning of the P2P network development and research, Napster and Gnutella determined the start. The former showed an approach with central elements within a single point of failure. The latter was unstructured and did not scale very well. An introduction of super peers made the unstructured network semi-structured [4]. A super peer is able to do more than other peers in terms of resources or network properties. The most recent research focuses on structured networks. With structured networks [8], [9], [10], such as distributed hash tables (DHTs), resources can be found in $O(\log n)$, where n is the number of peers in the system. [3]

2. Description of Work

The key problem today is the use of NATs in a P2P environment. Thus the question to solve can be formulated as: Which type of functions and mechanisms are required to define a P2P network,

which can transparently traverse NATs? Designated peers have to be chosen as a kind of a super peer that opens NAT ports. An example application, such as a chat client, has to be used to demonstrate in a P2P manner the practical applicability of the scheme developed.

The following technical requirements should be met:

- Although the framework designed should be open for any P2P system, only a DHT implementation should be developed.
- The Java version should be at least Java 1.4. Instead of blocking IO (java.io), the non-blocking new IO (java.nio) [5] should be used to achieve a high performance communication.
- The communication protocol should be designed in a XML format.
- Use available design/implementation frameworks and libraries where possible. For example, do not implement a new XML parser. Identify the features of these libraries and find out if the library can be used. If necessary, adapt these libraries if possible.
- Use JavaDoc to document the code. Optionally, use JUnit [7] or similar tools to test those methods while developing.
- Use a GPL license [6] for the code.

3. Thesis Goals

- The main target consists of developing a framework for a P2P system in support of traversing NATs transparently. The framework shall show the following characteristics:
 - Universal design: any P2P network can be implemented. The design should be flexible and extendible. A DHT should be integrated into the framework.
 - Support for NAT traversal.
 - A sample application on top of the network should be implemented.

4. Activities

Based on the description of work the following activities — some in parallel — are recommended. Based on the time schedule to be defined initially, a reordering and a more detailed completion is required.

- Start with an introduction to fields of interest and provide a motivation for your work. The focus has to be set to application fields where P2P networks are more suitable than centralized approaches. This includes also an outline of applied and applicable concepts and terms.
- Provide an overview of the state-of-the-art with respect to related work in the areas of:
 - P2P networks, different approaches and different concepts.
 - P2P frameworks. List all available frameworks and compare the framework with your approach.
 - NAT traversal / STUN.
- Out of the motivation and the analysis of related work, derive requirements of the prototype. Describe the high level (user point of view) and the low level (system point of view) requirements.

- Discuss and list advantages and drawbacks of your prototype with your sample application using a centralized approach and a P2P approach. Additionally, discuss the security aspect in your prototype in a fully decentralized approach.
- Design your framework and determine relevant schemes and mechanisms.
- Propose and design the prototype of the framework and define the general interface of P2P networks.
- Find free (compatible to GPL) libraries, that can be used for the prototype. Decide whether it is more efficient to adapt the library or write your own library.
- Implement the prototype. Build the framework as flexible as possible. Use open and flexible interfaces where appropriate and allow for an extensible implementation.
- Perform a benchmark showing that the prototype can handle many/hundreds of connections. Optionally, compare the benchmark with a threaded blocking approach. Validate the requirements and draw final conclusions.

5. General Notes

- Provide a proposed schedule for your steps within the first two weeks of your work. Clarify details with your supervisor and finalize the schedule of tasks, basically in a bi-weekly fashion. Prepare an intermediate report and a short internal presentation after half time (date to be set) and discuss this with your supervisor. At the thesis' end a final public presentation has to be given.
- The final written report will document all work undertaken and remember that this report must be self-contained. Major technical basics are to be included and detailed knowledge obtained during the work must be documented. Assumptions, design decisions, configuration choices, and results are part of the report as well as implementation details and usage information. Correct bibliographic references and a list of papers, recommendations, and descriptions used must be added, including those ones given below.
- Establish weekly meetings with the supervisor to report on progress and problems.
- Students involved are required to read and answer e-mails related to this project at least twice a week.
- The sheet on hints for Diploma/Students Thesis work is required to be known.

6. Formal Results

Besides an oral presentation of your work in exactly 20 min plus 10 min for questions and answers during the public presentation, the following written documents are part of your work and need to be handed in to the supervisor in time:

- A report in a soft cover binding and in 3 copies (in English, preferred): It covers the problem to be solved, the discussion of the design choices, a set of arguments on the final design choice, a list of solved and open issues, a table of content and figures (including tables), a valid list of bibliographic references, and optional appendices as required. A critical consideration of the task, the work, and the result will conclude the report. The official acknowledgement section is mandatory, a personal one optional, however recommended, as usually a number of people took part in the process of finalizing the thesis. The text processing shall be done in FrameMaker (preferred).
- A documentation of the implemented/configured system, covering the systems view point, an implementation description, a use and installation manual, a documentation of all program and data structures developed or utilized. All of this may be part of the report below, however, in case it will not be included, it must exist in a separate document.

- A dedicated CD has to be produced: (1) a collection of all program sources, software components, protocol implementations utilized, hardware/product documentation in PDF, related work papers in PDF or full HTML pages, and a directory description; (2) the written thesis (report) in source files, figures in source file and gif or eps, and a full printable PS as well as PDF file, the set of slides for the final presentation in PowerPoint, and all further material used, if available in electronic form, such as all existing and documented test scenarios, testing plans, and test results.
- A German (or English, in case of a German report) summary of maximum 2 pages, which will enable a quick and clear survey of this work's tasks and results. This summary will be part of the bound report, and is included after the front page and before any other text will follow. It includes four short sections on: 1. Introduction, 2. Aims and Goals, 3. Results, and 4. Further Work.
- The complete set of copies of the report, the CD, and the talk must be completed and handed in to the supervisor in time before the diploma thesis will change into "submitted" status.

7. References

The following list of references addresses key points. Further paper, scenario, and document research is a must:

- [1] jxta.org, URL: <http://www.jxta.org/>, last visited: 30.09.2005
- [2] Fedora Project, URL: <http://fedora.redhat.com/>, last visited: 30.09.2005
- [3] Hari Balakrishnan and M. Frans Kaashoek and David Karger and Robert Morris and Ion Stoica, *Looking up data in P2P systems*, ACM Press, 2003
- [4] Yatin Chawathe and Sylvia Ratnasamy and Lee Breslau and Nick Lanham and Scott Shenker, *Making Gnutella-like P2P Systems Scalable*, In Proceedings of the ACM SIGCOMM '03 Conference, Karlsruhe, Germany, August, 2003
- [5] Michael T. Nygard, *Master Merlin's new I/O classes*, URL: <http://www.javaworld.com/javaworld/jw-09-2001/jw-0907-merlin.html>, last visited: 30.09.2005
- [6] GNU General Public License, URL: <http://www.gnu.org/copyleft/gpl.html>, last visited: 30.09.2005
- [7] JUnit, Testing Resources for Extreme Programming, URL: <http://www.junit.org/index.htm>, last visited: 30.09.2005
- [8] P. Maymounkov and D. Mazieres. *Kademlia: A peer-to-peer information system based on the xor metric*. 1st International Workshop on Peer to Peer Systems (IPTPS'02), Boston, MA, USA, March 2002.
- [9] S. Ratnasamy, P. Francis, M. Handley, R. Karp, S. Shenker. *A scalable content-addressable network*. In Proc. ACM SIGCOMM'01, San Diego, CA, USA, August 2001.
- [10] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, H. Balakrishnan. *Chord: A scalable peer-to-peer lookup service for internet applications*. ACM SIGCOMM, San Diego, March 2001, pp 149-160.