Peer-to-Peer Issue Tracking System: Challenges and Solutions

Vijay Srinivas Agneeswaran[*], Rammohan Narendula[†] and Karl Aberer[‡]
Distributed Information Systems Lab (LSIR)
Ecole Polytechnique Fédérale de Lausanne.

Issue tracking[1] tools such as Jira or Bugzilla have become important for software development, especially with distributed development teams becoming the norm in several organizations. It is interesting to see that some open source software come with their own issue tracking systems, such as Eclipse bugzilla (https://bugs.eclipse.org/bugs/). However, these tools are centralized and are project specific. The software development community needs a decentralized, project independent issue tracking tool that enables the community of users to share and have access to knowledge.

By centralized, we mean that the existing issue tracking systems are managed centrally - with centralized control. Moreover, they are also hosted on centralized servers. However, both these may not be attractive for the open source community. First, they would love to do away with centralized systems for autonomy reasons. Most users would love to preserve the autonomy of their data and to be able to dictate how it must be shared over the network. Secondly, the centralized repositories may face scalability and fault-tolerance issues with increasing number of clients or users and their geographic spread - mainly due to the fact that bug-tracking or issue tracking in general is a data intensive task. Another key motivation for the work is that knowledge management in software engineering must occur across projects - a problem faced by a software developer (say in debugging or in developing a particular plugin for a particular software) in a particular project may be faced

[*]vijaysrinivas.agneeswaran@epfl.ch
[†]rammohan.narendula@epfl.ch
[‡]karl.aberer@epfl.ch

[1]Issue tracking systems can be seen as a generalization of bug reporting systems.

repeatedly by other developers across the community. Hence, the need for a decentralized autonomous issue tracking tool that would enable software engineering knowledge to be stored, accessed by a geographically dispersed community.

In this position paper, we identify the key challenges of building a Peer-to-Peer (P2P) issue tracking system and outline the initial steps we have taken in this direction. A first question that can be easily posed is why should there be a P2P solution and why a simple decentralized solution is not sufficient. The answer lies in why P2P systems are actually becoming widely used (not for scalability or fault-tolerance) - autonomy and heterogeneity. Users would like to control how their data must be shared (selectively). Different nodes (used by the users) may use different formats to represent the issue tracking data - leading to data heterogeneity. Data heterogeneity has been a key issue in the semantics web research [1]. The W3C has come with some standards for the same, including the use of Resource Description Framework (RDF) for data representation.

RDF stores such as Jena [2] and others allow RDF elements to be stored in relational database and allow queries over the RDF data, in the form of SPARQL [2] or RDQL[3] queries. Many existing RDF stores store RDF data in centralized repositories, making it difficult to provide scalability and fault-tolerance. Moreover, completely decentralized approaches for data management are required, for reasons of freshness and flexibility [3].

The key challenges we identify in building a P2P issue tracking system are:

- Scalability: The inability of the infrastructure to degrade gracefully when the storage, network or processing load grows has impaired such efforts in the past.

- Semantic heterogeneity: Different sources store data in different formats or by using different schemas.

- Security: Users must be sure that the data is being shared the way they have specified it. Hence, the need for access control policies and their enforcement in a purely decentralized or P2P setting is very important.

[2]http://www.w3.org/TR/rdf-sparql-query/
[3]http://www.w3.org/Submission/RDQL/

To summarize, a P2P secure RDF store is the platform we need to build the P2P issue tracking system we visualize. GridVine is an effort that has been made in this regard by us. GridVine [4] is a system that addresses both scalability and semantic heterogeneity, achieving data sharing in large scale P2P systems. GridVine facilitates RDF triple based queries that allow users to retrieve structured content from a P2P system. We are currently building an access control framework over GridVine that would enable users to specify how the data must be shared and enforce it in the P2P network.

Structured p2p networks are characterized by the fact that resources are stored at the end nodes- the owners, and the index of the shared items is stored with in the p2p network in the form of DHT enabling an efficient search. However, the wide adaptability of the p2p data storage systems is hindered by the fact that virutally all the index information and the resources are exposed to every member of the network. For the systems to mature and evolve towards a replacement for traditional client-server data sharing systems, efficient access control mechanisms should be enforced to control the access of the resources by the peers.

The possible points for enforcing the access control policy are the network itself and the end nodes (typically the owners). We argue that any robust p2p information system should employ the enforcement at both the places, which means controlling the access to *both* indexes and resources in a structured p2p information system. Only authorized users should be able to get results for the searches they do based on keywords. These results may include just the extra meta information associated with the resource as part of the index stored in the DHT or the resource itself. Before allowing a user to access the physical resource, the access control policy is checked and request is granted or denied based on the policy statement associated with the user.

The PHera system proposed in [5] deals with access control in super peer based p2p infrastructures. And the protocol trusts all the super peer nodes to enforce the access control policy. Sub-peers stores their access control policy information with the super peers. This solution can not be applied to the structured p2p networks. We present some research directions and brief overview of the current solution we are developing. We assume the presence of a secured communication in place, which is free from eavesdropping, traffic analysis by unintended participants in the network. Secured Socket Layer (SSL) based p2p infrastrucutres are already proposed for this purpose [6], [7].

We propose that access control on the index can be done by either a

Controlled Queries approach or a *Controlled Replies* approach. In controlled queries approach, we ensure that only intended users can post queries for the resources. This can be done in several ways and one of them is by obfuscating the *key* used to search in the system. For example, instead of publishing a resource with name *myReport.doc*, the user publishes with a identifier generated using a *secret key*. So a user who does not have access to this key can not derive mappings between the keywords he is interested in and the actual keys used in the system. A group of users decide the key out-of-band for this purpose. So in such a system, ability to query the system with the right key word itself authorizes the user.

In controlling replies approach, the philosophy is to reply to queries that originate only from the intended users. Access control policy information is stored with the index too and is checked for each request for taking a decision on the request. To enforce this technique, we propose the concept of *Trusted Sub Overlay (TSO)*. Each TSO is an overlay on top of existing overlay consisting of only trusted nodes for enforcing the access control function. All the resources for which the index is stored in a TSO are identified with a `(TSO_ID,resource_ID)` pair. The resources for which the index is stored in the normal overlay are referred with a single identifier. In this model, a node plays two roles one as member of the overlay and member of TSO, hence it is responsible for its identifier space as part of the overlay and that of the TSO. Each TSO has its own complete identifier space.

To search in the public overlay, a search query with just the corresponding resource id is issued and the responsible peer for that id responds. To search in the TSO, a search query with TSOID and resource ID is issued, which is forwarded to one of the nodes in the corresponding TSOID, which in turn forwards the query to the responsible peer for that ID in that TSO. This can be be enabled in many ways. All the peers in the TSO inserts (TSO_ID, node_ID) pair into the public overlay.

References

[1] Nigel Shadbolt, Tim Berners-Lee, and Wendy Hall. The Semantic Web Revisited. *IEEE Intelligent Systems*, 21(3):96–101, 2006.

[2] Brian McBride. Jena: A Semantic Web Toolkit. *IEEE Internet Computing*, 6(6):55–59, 2002.

[3] Staab, Steffen and Stuckenschmidt, Heiner, editors. *Semantic Web and Peer-to-Peer*. Springer-Verlag, Berlin Heidelberg, 2006.

[4] Karl Aberer, Philippe Cudré-Mauroux, Manfred Hauswirth, and Tim Van Pelt. GridVine: Building Internet-Scale Semantic Overlay Networks. In Sheila A. McIlraith, Dimitris Plexousakis, and Frank van Harmelen, editors, *International Semantic Web Conference*, volume 3298 of *Lecture Notes in Computer Science*, pages 107–121. Springer, 2004.

[5] Bruno Crispo, Swaminathan Sivasubramanian, Pietro Mazzoleni, and Elisa Bertino. P-hera: Scalable fine-grained access control for p2p infrastructures. In *ICPADS '05: Proceedings of the 11th International Conference on Parallel and Distributed Systems (ICPADS'05)*, pages 585–591, Washington, DC, USA, 2005. IEEE Computer Society.

[6] Narendula Rammohan. A Note on Secure P-Grid. Technical Report LSIR-REPORT-2008-005, Swiss Federal Institute of Technology (EPFL), 2006.

[7] Daniel Brookshie, Darren Govoni, Navaneeth Krishnan, and Juan Carlos Soto. *JXTA: Java P2P Programming* . Sams; 1st edition (April 1, 2002), Sebastopol, CA, 2002.