Trustworthy Distribution and Retrieval of Information over HTTP and the Internet

Isaí Michel Lombera, Yung-Ting Chuang, Peter Michael Melliar-Smith, Louise E. Moser
Department of Electrical and Computer Engineering
University of California, Santa Barbara
Santa Barbara, CA 93106 USA
imichel@ece.ucsb.edu,ytchuang@ece.ucsb.edu,pmms@ece.ucsb.edu,moser@ece.ucsb.edu

***Abstract*— This paper describes a novel information distribution and retrieval system, named iTrust, that operates over HTTP and the Internet and that provides trustworthy access to information. iTrust is a completely distributed system, with no centralized mechanisms and no centralized control, that avoids subversion or censorship of information. Individuals submit information they wish to share to nodes on the Internet that distribute metadata to random participating nodes. Likewise, users submit requests containing metadata for information they wish to retrieve to random participating nodes. The paper presents an overview of the iTrust strategy, implementation, user interface, and performance. iTrust can effectively enable citizens to distribute and retrieve information over the Internet, even in the presence of subverted or non-operational nodes.**

***Keywords*-information distribution and retrieval; distributed search; trustworthy information access; citizen-centric service**

I. Introduction

Our trust in the accessibility of information over the Internet and the Web (hereafter referred to as the Internet) depends on benign and unbiased administration of centralized search engines and centralized search indexes. Unfortunately, the experience of history, and even of today, shows that we cannot depend on such administrators to remain benign and unbiased in the future.

To ensure that the distribution and retrieval of information over the Internet is not subverted or censored, alternative search mechanisms must be provided. The availability of multiple search engines is important, but that protection is weakened by the small number of search engines available today. An alternative to centralized search, an effective completely distributed search, without centralized mechanisms and without centralized control, is an important assurance to users of the Internet that a small number of administrators cannot prevent them from distributing their ideas and information, and from retrieving the ideas and information of others.

The thesis of this research is that the distribution and retrieval of information without centralized search engines and without centralized search indexes is a critical enabling technology for users to have trust in the Internet for access to information. It is important to ensure that such a trustworthy information distribution and retrieval system is available when it is needed, even though a user might normally use a conventional centralized search engine.

In this paper, we describe iTrust, a novel information distribution and retrieval system that operates over the HyperText Transfer Protocol (HTTP) and the Internet and that provides trustworthy access to information. Section II of the paper presents an overview of the iTrust strategy, Section III describes our implementation of iTrust based on HTTP, and Section IV describes the user interface. Performance results are presented in Section V. Related work, and the conclusion and future work, are presented in Sections VI and VII, respectively.

II. Overview of iTrust

The iTrust information distribution and retrieval system involves no centralized mechanisms and no centralized control. We refer to the nodes that participate in an iTrust network as the participating nodes or the membership. An iTrust network might correspond to participants with specific interests, or it might correspond to a social network. Multiple iTrust networks within the Internet may exist at any point in time, and a node may participate in several different iTrust networks at the same time.

In an iTrust network, some nodes (the source nodes) produce information, and make that information available to other participating nodes. The source nodes produce metadata that describes their information, and distribute that metadata to a subset of participating nodes that are chosen at random, as shown in Figure 1. The metadata are distinct from the information that they describe, and include a list of keywords and the URL of the source of the information.

Other nodes (the requesting nodes) request and retrieve information. Such nodes generate requests (also referred to as queries) that refer to the metadata, and distribute the requests to a subset of the participating nodes that are chosen at random, as shown in Figure 2.

The participating nodes compare the metadata in the requests (queries) they receive with the metadata that they hold. If such a node finds a match, which we call an encounter, the matching node returns the URL of the associated information to the requesting node. The requesting node then

 ISBN: 978-1-61208-141-0

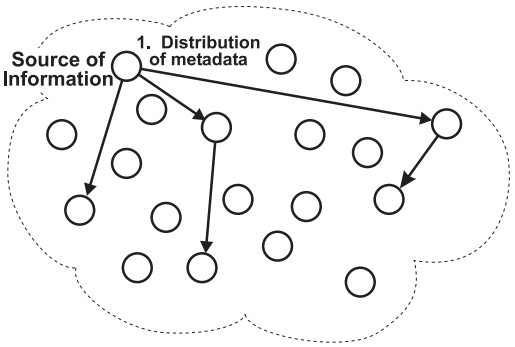

Figure 1. A node (a source node) distributes metadata, describing its information, to randomly selected nodes in the network.

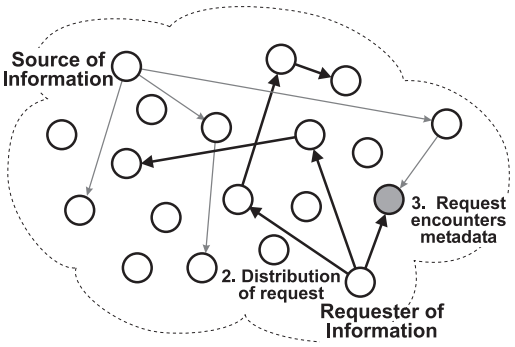

Figure 2. A node (a requesting node) distributes its request to randomly selected nodes in the network. One of the nodes has both the metadata and the request and, thus, an encounter occurs.

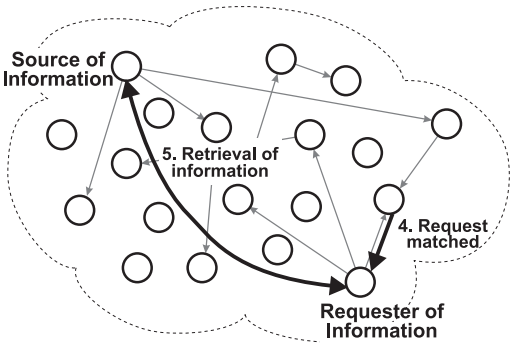

Figure 3. A node matches the metadata and the request and reports the match to the requester, which then retrieves the information from the source node.

uses the URL to retrieve the information from the source node, as shown in Figure 3.

Distribution of metadata and requests to relatively few nodes suffices to achieve a high probability that a match occurs. Moreover, the strategy is robust. Even if some of the randomly chosen nodes are subverted or non-operational, the probability of a match is high, as shown in Section V. Moreover, it is not easy for a small group of nodes to subvert the iTrust mechanisms to control which information is delivered and which is suppressed.

III. HTTP Implementation

The current implementation of iTrust uses HTTP to distribute metadata and requests. Each node is implemented using PHP (PHP: Hypertext Preprocessor) on an Apache Web server, thereby allowing any user with a Web browser on any platform to interact with the node. Node information is stored locally in an SQLite database. Multiple nodes can be installed on a single Web server by creating multiple virtual Web sites; multiple nodes on a single Web server have separate SQLite databases.

A. Membership

The membership list for a node is stored locally in an SQLite database table that contains node records whose fields are the node identifier and the node address. The node address may be either an IP address or a URL. A node is not verified when it is added to the membership as there is no guarantee that a node is always available. For example, a cell phone or a laptop with a WiFi connection may enter and leave its base station signal range multiple times a day. In practice, the only restriction on node addresses is that the Web site document root has Web server write permissions for saving uploaded resources.

B. Resources

Resources are files or groups of files uploaded to nodes in the membership. The list of resources on a node is stored in an SQLite database table that contains resource records whose fields are a resource handle, file path, and expiration date. The resource handle is a shortened random name (typically with 32-64 characters), which can be referenced across participating nodes. The file path is the name of the file on the local node disk. Thus, a participating node retrieving data may simply request a resource by the handle to the source node, instead of using the entire file path. The expiration date specifies when a given resource and associated keywords should be deleted. It allows time-sensitive information to be removed automatically from queries past the expiration date when the information is no longer relevant. For example, yesterday's weather forecast is not included in today's weather queries.

Resource files can be placed on a node using the Common Gateway Interface (CGI) by means of a file dialog, or through the cURL package provided by PHP. In the case of cURL, the file(s) is fetched directly by the node and written to the local disk.

When a resource is entered into the SQLite database, metadata for the resource file is generated using the Apache Tika/Lucene packages. These packages classify metadata based on content such as text strings, and file attributes such as data type, file size, *etc*. Also, the user may supply additional metadata keywords. The metadata keywords are stored in a SQLite database table, and the associations

 ISBN: 978-1-61208-141-0

between a resource and a keyword are stored in a separate table, thus normalizing the database.

C. Metadata distribution

Periodically, metadata keywords and other information about one (or more) resources on a node are collated and compiled into an XML file (also referred to as the metadata list) which describes the resource. The periodicity depends on the node and the platform; however, in practice, the user can update the metadata list at any time by clicking a button or running a cron job. The resource description in the metadata list includes the resource handle, file path, and expiration date for the resource. The metadata list includes all associated keywords for the resource.

After creation of the metadata list, random nodes in the membership are contacted and informed of a metadata update by means of an HTTP POST statement. The number of random nodes that are selected for metadata distribution is a tunable parameter in the iTrust node configuration file. The contact message includes the source node IP address and metadata list URL, which are stored on the receiving node. Each contacted node decides if and when to retrieve the metadata list. The retrieval period is receiving node dependent, so as not to trigger an instant download of the metadata list file by multiple nodes.

If a retrieval occurs, the receiving node retrieves the metadata list file from the source node and processes the XML. The metadata list is stored on the receiving node. If there are multiple resources represented by sets of metadata in the metadata list, then processing continues on the next set of metadata, until the end of the metadata list is reached. XML processing is performed using the SimpleXML PHP extension.

D. Query relaying

Search queries (requests) are the main interaction between the user and an iTrust node and, as such, require the most processing. A search query originates at a single node, but the query message may be relayed among multiple nodes in the iTrust network. A query message may take any available network path with the sole restriction that a node never relays the same query twice.

The query field is a simple HTML form text box on the current node; the command to begin the query is detection of pressing a submit button or enter key. The query text itself is URL encoded to facilitate later operations; no custom processing on the query text (*e.g.*, duplication detection, grammar checking, *etc.*) is performed.

Two additional variables are created before a node sends the query: the node IP address and the query identifier. The node IP address is read from the iTrust configuration file; it ensures that queried nodes know which node originally sent the query. The query identifier ensures that no query is relayed twice and helps manage multiple queries sent from a single node.

Once the query is ready to be sent, multiple random nodes are selected from the node database table and the query is packaged into an HTTP POST statement. The frequency of node selection is another customizable iTrust configuration variable and need not be the same at different nodes. Node selection for querying is not necessarily the same as node selection for metadata list distribution; both variables may be set by the administrator of the node.

A node sends the HTTP POST statement to random nodes, and each node (both sender and receivers) saves a copy of the query before processing. If any text in the query matches the metadata keywords, an encounter occurs. The queried node then sends an HTTP POST response back to the originating node. The originating node is obtained from the sender node's IP address given in the query package. The response includes the query identifier (again obtained from the query package), the encountered node's IP address (hereafter referred to as the source node), and the resource handle of the matching resource. Additionally, the querying node saves the response in an SQLite database table for later processing.

A queried node, regardless of whether or not it has an encounter, may relay the query as if it were the originating query node. The only difference is that it does not recreate the two additional variables (source IP address and query identifier); those variables are relayed without modification. However, before relaying the query, the node checks the query identifier to ensure that the node has not already seen the query. If the node has already saved the query identifier, it does not relay the message.

Note that the current node in this context may be either the node sending the query or the node receiving the query. In case the receiving node has not yet relayed the query, it relays the query to nodes randomly selected from the membership and records the query identifier. In case the receiving node has already relayed the query, the receiving node ignores the query. Because of the decentralized random nature of iTrust, the original node that sent the query might have the same query relayed back to itself. In this case too, the current node ignores the query.

After waiting for responses a certain amount of time, the querying node displays all of the source nodes with appropriate resource handles. All encounters are thus recorded on the querying node, and the user is informed of which nodes have resources matching the query.

E. Retrieving resources

All query results are recorded in the requesting node's SQLite database table, *i.e.*, the source node IP address and resource handle. When the user selects a query result, the source node is sent an HTTP GET statement with the resource handle, and the source node returns the resource

 ISBN: 978-1-61208-141-0

file directly to the user. Alternatively, the source address and resource handle can be encoded directly into a URL; the user then accesses the file using an HTML anchor tag.

IV. User Interface

The iTrust user interface is a Web-based interface where the user can both administer and query the nodes through Web pages. Query results from multiple nodes are presented in a single Web page following a query. Node administration and user queries are separated into distinct Web pages to keep tasks distinct and easily manageable.

A. Node administration

The user may change the membership, add source nodes, distribute metadata, and perform other administration tasks through the administration interface shown in Figure 4.

A node is added to the membership by entering the node IP address or URL on a comma delimited list inside an HTML form text box. Double listing is not permitted; duplicates are removed from the list. However, multiple nodes are permitted as long as the Web site document root is distinct (*e.g.*, both www.example.com/foo and www.example.com/bar are allowed).

Figure 5 shows the resource insertion Web page. A resource is added to a node by means of an HTML form file control; this control permits the user to upload a file from his/her local machine. Alternately, a Web site URL can be specified, and the node then fetches the contents at that URL. The uploaded contents are post-processed, using the Apache Tika/Lucene package, to generate descriptive metadata (*i.e.*, keywords) automatically. The user can customize several parameters for metadata creation, including indexing by file raw content (literal text strings) or file meta content (file size, type, *etc.*). In addition to automatic metadata creation for an uploaded resource, the user may add new keywords or remove existing keywords. Finally, the user may assign an expiration date to the resource.

Administration tasks also include file administration functions to allow the user to setup, restore, or reset iTrust nodes easily. Clearing the membership, deleting all resources and metadata associations, and resetting a node to its initial setup state can all be done with a single button click. The task of pushing all metadata changes to random nodes is also accomplished with a single button click.

B. User queries

The user may perform queries, view the query results, and obtain resources through the user interface.

Querying is done through a single HTML form text box, whereupon the query is registered on the node and distributed throughout the iTrust network. The user is shown a status/wait Web page while the query is relayed among nodes; a result Web page is shown after a wait page timeout. The default timeout is 3 seconds and, thus, a query incurs a 3 second latency between initialization of the query and display of the query results. However, the wait page timeout is also configurable by the node administrator.

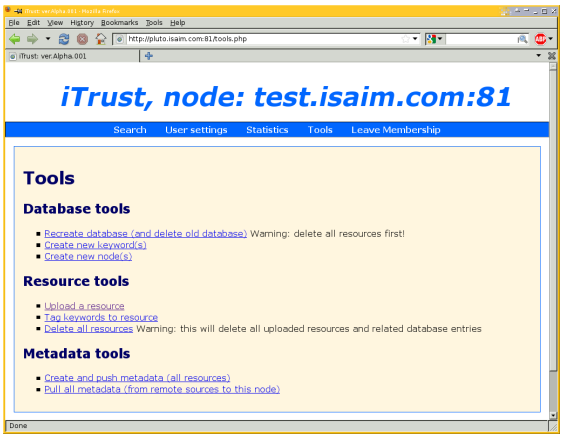

Figure 4. The administration interface.

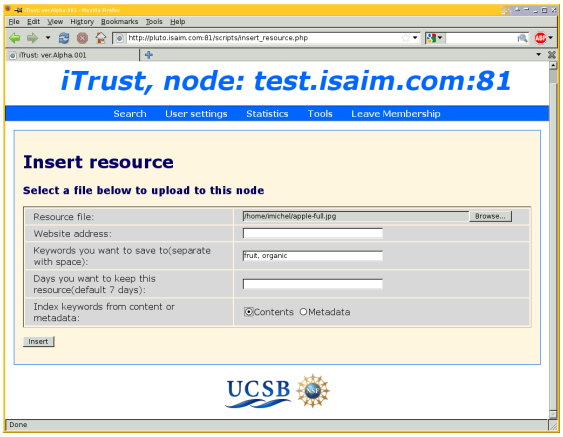

Figure 5. The insert resource Web page.

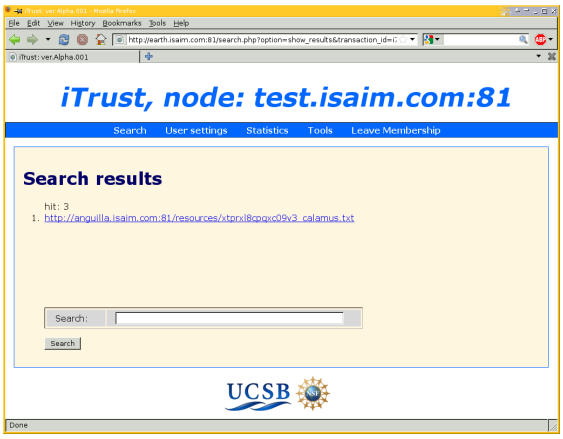

Figure 6. The query results Web page.

Figure 6 shows the query results displayed on a new Web page (the wait page automatically redirects to the new page)

in a simple HTML list. Each encounter is shown as a list item with the source address and resource handle encoded into a single URL.

The user may click the URL to retrieve the resource file; the format of the file is the originally uploaded format (there is no MIME-type modification). If the Web browser recognizes the file type, it handles the data accordingly; otherwise, it calls the operating system to open the data file.

C. User settings

For querying, the three primary user settings (which the user sets on the user settings page) are the number of nodes to which the metadata are distributed, the number of nodes to which the requests are distributed, and the search duration.

The number of nodes to which the metadata are distributed and the number of nodes to which the requests are distributed must, of course, be less than the number of participating nodes in the membership.

The search duration refers to the lifetime that a search query exists. The user may specify how many days a query will be stored in the database. When a user initiates a query, the system adds its creation time to the database. Later, when the user initiates a new query, the system checks and deletes expired queries from the database.

These user settings apply to the entire duration of a search session. The search session starts when a user accesses the search Web page and ends when the user exits the browser window or tab. The PHP session functions are used to automate this process.

V. PERFORMANCE EVALUATION

If a node receives a request and it holds metadata that matches the request, we say that the node has a match. In the performance evaluation, we consider the probability of a match, using both analysis and simulation based on our HTTP implementation. We assume that all of the participating nodes have the same membership set. In addition, we assume that the Internet is reliable and that all of the participating nodes have enough memory to store the source files and the metadata.

A. Analysis

In an iTrust network with a membership of n nodes, we distribute the metadata to m nodes and the requests to r nodes. The probability p that a node has a match then is:

$$p = 1 - \left(\frac{n-m}{n}\right)\left(\frac{n-m-1}{n-1}\right)\cdots\left(\frac{n-m-r+1}{n-r+1}\right) \quad (1)$$

Formula 1 holds for $n \geq m + r$. If $m + r > n$, then $p = 1$.

As above, we distribute the metadata to m nodes and the requests to r nodes in an iTrust network with a membership of n nodes. But now we introduce another variable x, which represents the proportion of the n nodes that are operational. In an iTrust network with a membership of n nodes, where x nodes are operational, the probability p that a node has a match is:

$$p = 1 - \left(\frac{n-mx}{n}\right)\left(\frac{n-mx-1}{n-1}\right)\cdots\left(\frac{n-mx-r+1}{n-r+1}\right) \quad (2)$$

Formula 2 holds for $n \geq mx + r$. If $mx + r > n$, then $p = 1$.

Figures 7, 8, and 9 show the probability p of a match obtained from Formulas 1 and 2 with $n = 72$ nodes where $x = 100\%$, 80%, and 60% of the participating nodes are operational, respectively, as a function of $m = r$ (the number of nodes to which the metadata and requests are distributed). As we see from the graphs, the probability p of a match increases and asymptotically approaches 1, as $m = r$ increases.

B. Simulation

Using our HTTP implementation described in Section III, we performed simulation experiments to validate the analytical formulas given above. In our simulation, we used libCURL (which is a free client-side URL transfer library for transferring data using various protocols) to collect the match probabilities.

Before we run our simulation program, we delete all resources and data from the SQLite databases. Next, the program adds all the nodes to the membership. Once all the nodes are added to the membership, we supply the number of nodes for distribution of metadata and requests, and the proportion of operational nodes, to the simulation program. Next, we call the source nodes to upload files and the program then creates the corresponding metadata. Then, the program randomly selects the nodes for metdata distribution and distributes the metadata to those nodes. Next, the program randomly selects the nodes for the requests and distributes the requests. If one or more nodes returns a response, there is a match and the simulation program returns 1; otherwise, there is no match and the simulation program returns 0.

Figures 7, 8, and 9 show the simulation results with 72 nodes where 100%, 80%, and 60% of the participating nodes are operational, respectively. As we see from these graphs, the simulation results are very close to the analytical results calculated from Formulas 1 and 2 where 100%, 80%, and 60% of the participating nodes are operational.

The lesson we learned from this performance evaluation is that iTrust retains significant utility even in the case where a substantial proportion of the nodes are non-operational.

VI. RELATED WORK

Today, centralized search engines are used commercially for Internet search, where metadata for the information are held in a centralized search index [2]. Requests are submitted to the central site, where they are matched against

 ISBN: 978-1-61208-141-0

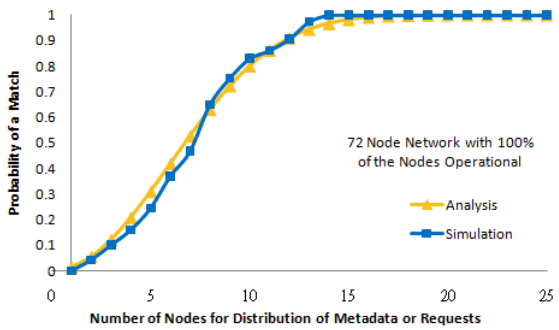

Figure 7. Match probability vs. number of nodes for distribution of metadata and requests in a network with 72 nodes where 100% of the nodes are operational.

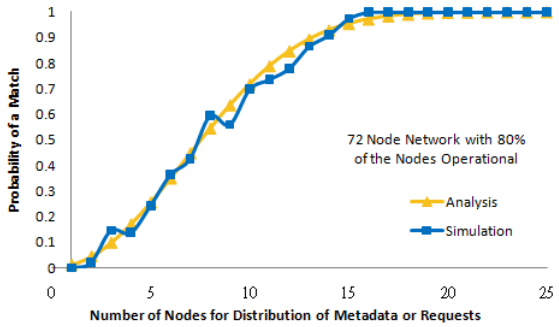

Figure 8. Match probability vs. number of nodes for distribution of metadata and requests in a network with 72 nodes where 80% of the nodes are operational.

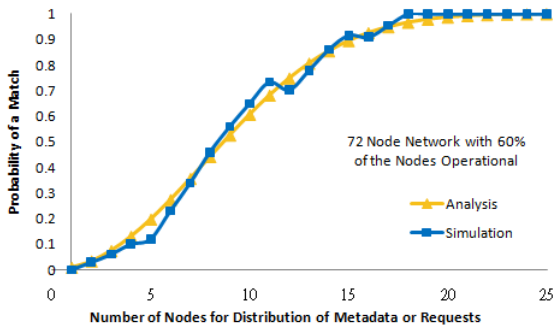

Figure 9. Match probability vs. number of nodes for distribution of metadata and requests in a network with 72 nodes where 60% of the nodes are operational.

the metadata keywords. Centralized search engines are efficient and scalable, but are vulnerable to manipulation by administrators. Similarly, the centralized publish/subscribe approach uses a centralized search index [5], where all of the information and all of the queries are published. The centralized publish/subscribe approach has the same issues of trust as the centralized search engine approach.

Mischke and Stiller [11] provide a taxonomy of distributed search mechanisms in peer-to-peer networks. Risson and Moors [12] provide another useful survey of distributed search mechanisms in such networks. The distributed publish/subscribe approach is categorized as either structured or unstructured.

The structured approach [1], [8], [10], [14] requires the nodes to be organized in an overlay structure, based on Distributed Hash Tables (DHTs), trees, rings, *etc.* The structured approach is more efficient than the unstructured approach, but it involves administrative control and additional overhead for constructing and maintaining the overlay network. Moreover, churn or malicious disruptions can break the structure.

The unstructured approach [4], [6], [7], [9], [15], [16], [17], [19] is typically based on gossiping, uses randomization, and requires the subscriber nodes and the publisher nodes to find each other by exchanging messages over existing links. The iTrust system uses the unstructured approach.

Gnutella [7] is the great grandfather of unstructured distributed search systems; it uses flooding of requests to find information. GIA [3] is an unstructured Gnutella-like peer-to-peer system that combines biased random walks with one-hop data replication to make search more scalable. Likewise, Sarshar *et al.* [13] combine random walk data replication with a two-phase query scheme in a Gnutella-like network for scalability. Yang and Garcia-Molina [18] use supernodes to improve efficiency, but reintroduce some of the trust risks of centralized strategies in doing so.

Freenet [4] is a more sophisticated and efficient system than Gnutella, because it learns from previous requests. In Freenet, nodes that successfully respond to requests receive more metadata and more requests. Thus, it is easy for a group of untrustworthy nodes to conspire together to gather most of the searches into their group, making Freenet vulnerable to subversion.

Ferreira *et al.* [6] use a random-walk strategy in an unstructured network to replicate both queries and data. BubbleStorm [15] is a probabilistic system for unstructured peer-to-peer search that replicates both queries and data, and combines random walks with flooding. Pub-2-Sub [16] is a publish/subscribe service for unstructured peer-to-peer networks of cooperative nodes, that uses directed routing (instead of gossiping) to distribute subscription and publication messages to the nodes. None of the above unstructured systems is particularly concerned with trust, as iTrust is.

Two other systems that, like iTrust, are concerned with trust are Quasar and OneSwarm. Quasar [17] is a probabilistic publish/subscribe system for social networks with many social groups. The authors note that “an unwarranted amount of trust is placed on these centralized systems to not reveal or take advantage of sensitive information.” iTrust does not use a structured overlay, and has a different trust objective than Quasar. OneSwarm [9] is a peer-to-peer data sharing system that allows data to be shared either publicly or anonymously, using a combination of trusted and untrusted

 ISBN: 978-1-61208-141-0

peers. OneSwarm is part of an effort to provide an alternative to cloud computing that does not depend on centralized trust. Its initial goal is to protect the privacy of the users; iTrust does not aim to conceal the users like OneSwarm does.

VII. Conclusion and Future Work

We have described iTrust, a novel information distribution and retrieval system with no centralized mechanisms and no centralized control. iTrust involves distribution of metadata and requests, matching of requests and metadata, and retrieval of information corresponding to the metadata. We have shown that, with iTrust, the probability of matching a query is high even if some of the participating nodes are subverted or non-operational. The iTrust system is particularly valuable for individuals or citizens who wish to share information, without having to worry about subversion or censorship of information.

In the future, we plan to evaluate the ease of installation and use of iTrust with various user populations and also to evaluate its reliability and efficiency in PlanetLab. We also plan to investigate the scalability of the iTrust system to thousands of nodes and, then, to extrapolate those results to millions of nodes. In addition, we plan to investigate a range of possible attacks on iTrust and countermeasures to such attacks. Our objective for iTrust is a network in which individual nodes can detect a potential attack, and can adapt to an attack to maintain trustworthy information distribution and retrieval even when under attack.

Acknowledgment

This research was supported in part by U.S. National Science Foundation grant number NSF CNS 10-16193.

References

[1] S. Bianchi, P. Felber and M. Gradinariu, "Content-based publish/subscribe using distributed r-trees," *Proceedings of Euro-Par*, Rennes, France, August 2007, pp. 537–548.

[2] S. Brin and L. Page, "The anatomy of a large-scale hypertextual Web search engine," *Proceedings of the 7th International Conference on the World Wide Web*, Brisbane, Australia, April 1998, pp. 107–117.

[3] Y. Chawathe, S. Ratnasamy, L. Breslau, N. Lanham and S. Shenker, "Making Gnutella-like P2P systems scalable," *Proceedings of the ACM SIGCOMM Applications Technologies, Architectures and Protocols for Computer Communications Conference*, Karlsruhe, Germany, August 2003, pp. 407–418.

[4] I. Clarke, O. Sandberg, B. Wiley and T. Hong, "Freenet: A distributed anonymous information storage and retrieval system," *Proceedings of the Workshop on Design Issues in Anonymity and Unobservability*, Lecture Notes in Computer Science, Berkeley, CA, July 2000, pp. 46–66.

[5] P. T. Eugster, P. A. Felber, R. Guerraoui and A. M. Kermarrec, "The many faces of publish/subscribe," *ACM Computing Surveys* 35:2, June 2003, pp. 114–131.

[6] R. A. Ferreira, M. K. Ramanathan, A. Awan, A. Grama and S. Jagannathan, "Search with probabilistic guarantees in unstructured peer-to-peer networks," *Proceedings of the Fifth IEEE International Conference on Peer-to-Peer Computing*, Konstanz, Germany, August 2005, pp. 165–172.

[7] Gnutella, http://gnutella.wego.com/

[8] A. Gupta, O. D. Sahin, D. Agrawal and A. El Abbadi, "Meghdoot: Content-based publish/subscribe over P2P networks," *Proceedings of the 5th ACM/IFIP/USENIX International Middleware Conference*, Toronto, Canada, 2004, pp. 254–273.

[9] T. Isdal, M. Piatek, A. Krishnamurthy and T. Anderson, "Privacy preserving P2P data sharing with OneSwarm," Technical Report UW-CSE, Department of Computer Science, University of Washington, 2009.

[10] J. Li, B. Loo, J. Hellerstein, F. Kaashoek, D. Karger and R. Morris, "On the feasibility of peer-to-peer Web indexing and search," *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems*, Lecture Notes in Computer Science 1735, 2003, pp. 207–215.

[11] J. Mischke and B. Stiller, "A methodology for the design of distributed search in P2P middleware," *IEEE Network* 18:1, January 2004, pp. 30–37.

[12] J. Risson and T. Moors, "Survey of research towards robust peer-to-peer networks: Search methods," Technical Report UNSW-EE-P2P-1-1, University of New South Wales, September 2007, RFC 4981, http://tools.ietf.org/html/rfc4981

[13] N. Sarshar, P. O. Boykin and V. P. Roychowdhury, "Percolation search in power law networks: Making unstructured peer-to-peer networks scalable," *Proceedings of the 4th International Conference on Peer-to-Peer Computing*, Zurich, Switzerland, August 2004, pp. 2–9.

[14] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek and H. Balakrishnan, "Chord: A scalable peer-to-peer lookup service for Internet applications," *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures and Protocols for Computer Communications*, San Diego, CA, August 2001, pp. 149–160.

[15] W. W. Terpstra, J. Kangasharju, C. Leng and A. P. Buchman, "BubbleStorm: Resilient, probabilistic, and exhaustive peer-to-peer search," *Proceedings of the ACM SIGCOMM Conference on Applications, Technologies, Architectures and Protocols for Computer Communications*, Kyoto, Japan, August 2007, pp. 49–60.

[16] D. A. Tran and C. Pham, "Enabling content-based publish/subscribe services in cooperative P2P networks," *Computer Networks* 52:11, August 2010, pp. 1739-1749.

[17] B. Wong and S. Guha, "Quasar: A probabilistic publish-subscribe system for social networks," *Proceedings of the 7th International Workshop on Peer-to-Peer Systems*, Tampa Bay, FL, February 2008.

[18] B. Yang and H. Garcia-Molina, "Improving search in peer-to-peer networks," *Proceedings of the 22nd IEEE International Conference on Distributed Computing Systems*, Vienna, Austria, July 2002, pp. 5–14.

[19] M. Zhong and K. Shen, "Popularity-biased random walks for peer-to-peer search under the square-root principle," Lecture Notes in Computer Science 4490, 2007, pp. 877–880.

 ISBN: 978-1-61208-141-0