Storm: Using P2P to Make the Desktop Part of the Web

Benja Fallenstein, Tuomas J. Lukka and Hermanni Hyytiälä

Hyperstructure Group
Agora Center, University of Jyv¨askyl¨a
P.O. Box 35, FIN-40014 Jyv¨askyl¨a

b.fallenstein@gmx.de, lukka@iki.fi, hemppah@cc.jyu.fi

Toni Alatalo
Department of Information Processing Sciences
University of Oulu, Finland
P.O.Box 3000, FIN-90100 Oulu

antont@oulu.fi

ABSTRACT

We present Storm, a storage system which unifies the desktop and the public network, making Web links between desktop documents more practical. Storm assigns each document a permanent unique URI when it is created. Using peer-to-peer technology, we can locate documents even though our URIs do not include location information. Links continue to work unchanged when documents are emailed or published on the network. We have extended KDE to understand Storm URIs. Other systems such as GNU Emacs are able to use Storm through an HTTP gateway.

Categories and Subject Descriptors: H.3.4 [Information Storage and Retrieval]: Systems and Software—*distributed systems, information networks*

General Terms: Design, Reliability, Performance

Keywords: dangling links, location-independent identifiers, content addressable networks, peer-to-peer

1. INTRODUCTION

While many Web pages are densely interlinked today, hypermedia functionality is still only rarely used in desktop documents. Hypermedia on the desktop would be useful for keeping hyperlinked notes, for referring to pieces of documents in discussion, or to refer from a short memo to a document with more in-depth information. Even though documents written with Microsoft Word or OpenOffice.org can nowadays be linked just like HTML files, users almost never seem to do so.

A major obstacle to hypermedia on the desktop are broken links. A user receiving hyperlinked documents by email, for example, would have to reconstruct the original directory structure; otherwise, the links will not work. Many hypermedia systems assume that identifiers either expose where a document can be found on the network, or they cannot be efficiently resolved on a global scale. Indeed, Berners-Lee [2] argues that using unique and random-looking numbers to identify documents is not possible on a global scale.

However, recent developments in peer-to-peer systems have rendered this assumption obsolete. Distributed hashtables (DHTs) and other content-addressable network algorithms [6, 8, 9, 10, 12, 15] allow location-independent identifiers to be resolved on a global scale. The use of peer-to-peer technology in hypermedia systems is the topic of an ongoing discussion in the hypermedia community [3, 7, 13, 14].

HT'03, August 26–30, 2003, Nottingham, United Kingdom.
ACM 1-58113-704-4/03/0008.

Balakrishnan et al. [1] argue that DHTs are suitable for the next generation Reference Resolution Services (RRS)[1]. DHTs can not only be used to efficiently retrieve documents, but also metadata *about* documents. For example, unlike many link services that only query a limited set of servers for links (e.g., [5]), all published external links to a document can be found efficiently using a DHT.

Location-independent identifiers enable non-breaking links on the desktop and the Web: When a document is published on the Web, it will have the same URI as on the author's local desktop; when a link to a document is emailed, the receiver can follow the link as long as their software can locate the linked-to document– on its local harddisk, attached to another email, on a work group server or on the public Internet. In Freenet [4], an anonymous P2P publication system, *cryptographic hashes*[2] are used as location-independent identifiers of document versions.

This paper describes Storm, a library for storing and retrieving data using location-independent URIs.

2. STORM

Storm (for STORage Module) is a library for both local and distributed storage and publishing. On the lowest level, data is stored in *blocks*, which are immutable byte sequences. As opposed to file systems, Freenet-like cryptographic identifiers are used. This may seem like a bad tradeoff: while these identifiers enable secure location-independent references, they do not allow the identified documents to change and are also less comprehensible to humans. However, these downsides are dealt with by the higher abstraction levels of Storm.

One of the key innovations here is the use of *pointer blocks* to build mutable documents on top of the lowest layer append-and-delete-only model. A pointer block is a Storm block which says, "The current version of document *X* is block *Y*. Pointer blocks *a,b*, specifying previous versions of this document, are now obsolete." Additionally, it contains a digital signature of the above.

The pointer *X* contains the public key used for signatures — this allows only the author of the document to validate new versions. Since a pointer block can override more than one older pointer block, this implements a nonlinear (branching) versioning model. More than one version of a document can be "current" (e.g.,

[1]The Domain Name System (DNS, RFC 1101) is an RRS system widely used on the Internet.

[2]A cryptographic hash is short bit-string (often 160 bits) computed from a document, so that it is practically impossible that two different published documents will ever have the same hash. For a 160 bit hash, hash collisions start to appear at about 2^{80} published documents; that's about 60 trillion per human living. See, e.g., [11].

when different people in a working group edit the document concurrently). There is also a scheme for storing only the differences between consecutive versions, but there is no space to explain it here.

Storm in itself is *not* a P2P system, but a data model and a library for using that data model with several different backends. Because of the simplicity of the lowest-level data model, it is able to use several different modes of networking, from simply copying blocks to using a full-fledged P2P system as a library — as of now, we are already successfully transferring blocks between Storm servers using the GISP DHT [6].

3. INTERFACING

Storm uses an experimental URN namespace (Uniform Resource Name, a persistent kind of URI) and we are preparing the registration of a real namespace with IANA. The URN is constructed analogously to a data URL (RFC 2397) from the MIME content-type, using the hash of the data instead of the data itself.

At the moment, Storm URNs can be used as a replacement for both file names and HTTP URLs as document identifiers. Browsers and applications can access Storm either by using the Java library directly or through an HTTP gateway, which locates and returns a block given its Storm URN. Documents can be uploaded into Storm through the gateway, using HTTP PUT. Using the gateway, we have made the Konqueror web browser as well as Netscape Navigator 4 understand Storm URNs. Using our extension, any KDE application can load from and save to Storm URNs. KDE automatically uploads changed documents into Storm when the user hits "Save." The Emacs text editor does not understand Storm URNs yet, but can load and save documents in Storm using the HTTP gateway. OpenOffice.org can load documents through the HTTP gateway, but cannot save changes, since the gateway does not yet support WebDAV (HTTP-based distributed authoring and versioning).

The HTTP gateway also provides a Web interface to Storm, giving the user a list of all documents stored on their local hard disk, and allowing them to conduct searches over the titles of all documents published on the underlying P2P network. For HTML, the server is also able to dynamically insert backlinks: links to all other HTML pages on the network that link to *this* page.

The most important downside of using Storm in lieu of file names and URLs are the long, cryptic URIs which would ideally be handled through some user interface changes to the file selector dialog boxes.

4. CONCLUSIONS

We have presented Storm, a storage system unifying the namespaces for private and published documents, making hyperlinks between desktop documents more practical. While the core of the system has been in use for almost three years, the peer-to-peer subsystem has only recently been implemented. Currently, a number of systems can use Storm through an HTTP gateway or the KDE libraries. It can also be used to implement the more ambitious Xanalogical hypertext model[3] [7]. Storm is available as Free Software from `http://sv.gnu.org/projects/storm`.

The main new features over previous systems are the use of cryptographic identifiers for versioned hypermedia storage on the desktop; smooth integration with ordinary office applications; pointer blocks for nonlinear versioning; and the use of a DHT for finding external hypermedia links.

Storm is by no means finished: more work is needed on, e.g., versioning when a pointer is editable by more than one person. A practical improvement will be WebDAV support, allowing many additional systems to save data in Storm.

[3]Our implementation of Xanalogical storage is available at `http://sv.gnu.org/projects/alph`.

5. ACKNOWLEDGEMENTS

This work was funded by the Agora Center InBCT project. We would like to thank Jukka Honkela, Erno Kuusela, Harri Oinas-Kukkonen, and Sarah Stehlig for comments and discussions.

6. REFERENCES

[1] H. Balakrishnan, S. Shenker, and M. Walfish. Semantic-free referencing in linked distributed systems. In *Proc. IPTPS03*, February 2003.

[2] T. Berners-Lee. Axioms of web architecture 2: The myth of names and addresses. http://www.w3.org/DesignIssues/NameMyth.html, Dec. 1996.

[3] N. O. Bouvin. Open hypermedia in a peer-to-peer context. In *Proceedings of the thirteenth conference on Hypertext and hypermedia*, pages 138–139. ACM Press, 2002.

[4] I. Clarke, S. G. Miller, T. W. Hong, O. Sandberg, and B. Wiley. Protecting free expression online with Freenet. *IEEE Internet Computing*, 6(1):40–49, 2002.

[5] G. Hill and W. Hall. Extending the microcosm model to a distributed environment. In *Proceedings of the European Conference on Hypertext (ECHT'94)*, pages 32–40, 1994.

[6] D. Kato. GISP: Global information sharing protocol - a distributed index for peer-to-peer systems. In *Proc. P2P'02*, pages 65–73, 2002.

[7] T. J. Lukka and B. Fallenstein. Freenet-like guids for implementing xanalogical hypertext. In *Proc. Hypertext'02*, pages 194–195. ACM Press, 2002.

[8] D. Malkhi, M. Naor, and D. Ratajczak. Viceroy: a scalable and dynamic emulation of the butterfly. In *Proc. PODC 2002*, pages 183–192. ACM Press, 2002.

[9] P. Maymounkov and D. Maziéres. Kademlia: A Peer-to-peer Information System Based on the XOR metric. In *Proc. IPTPS02*, 2002.

[10] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Schenker. A scalable content-addressable network. In *Proc. SIGCOMM 2001*, pages 161–172. ACM Press, 2001.

[11] B. Schneier. *Applied Cryptography*. John Wiley & Sons, 1996.

[12] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proc. SIGCOMM 2001*, pages 149–160. ACM Press, 2001.

[13] M. K. Thompson and D. C. D. Roure. Hypermedia by coincidence. In *Proceedings of the twelfth ACM conference on Hypertext and Hypermedia*, pages 129–130. ACM Press, 2001.

[14] U. K. Wiil, N. O. Bouvin, D. Larsen, D. C. D. Roure, and M. K. Thompson. Peer-to-peer hypertext. In *Proceedings of the thirteenth conference on Hypertext and hypermedia*, pages 69–71. ACM Press, 2002.

[15] B. Y. Zhao, J. D. Kubiatowicz, and A. D. Joseph. Tapestry: An infrastructure for fault-tolerant wide-area location and routing. Technical Report UCB/CSD-01-1141, Computer Science Division, U. C. Berkeley, 2001.