File Management in a Mobile DHT-based P2P Environment

Khalid Ashraf[1], Rachid Anane[2] and Behzad Bordbar[1]
[1]*School of Computer Science, University of Birmingham*
{k.ashraf, b.brodbar}@cs.bham.ac.uk
[2]*Faculty of Engineering and Computing, Coventry University*
r.anane@coventry.ac.uk

ABSTRACT

The emergence of mobile P2P systems is largely due to the evolution of mobile devices into powerful information processing units. The relatively structured context that results from the mapping of mobile patterns of behaviour onto P2P models is however constrained by the vulnerabilities of P2P networks and the inherent limitations of mobile devices. Whilst the implementation of P2P models gives rise to security and reliability issues, the deployment of mobile devices is subject to efficiency constraints. This paper presents the development and deployment of a mobile P2P system based on distributed hash tables (DHT). The secure, reliable and efficient dispersal of files is taken as an application. Reliability was addressed by providing two methods for file dispersal: replication and erasure coding. Security constraints were catered for by incorporating an authentication mechanism and three encryption schemes. Lightweight versions of various algorithms were selected in order to attend to efficiency requirements.

Keywords: Peer-to-Peer systems, file dispersal, replication, erasure coding, security, reliability, efficiency.

1. Introduction

The evolution of mobile devices into powerful information units has been marked by the emergence of patterns of behaviour that display many of the characteristics of peer-to-peer (P2P) modes of interaction. The flexibility of these devices is due largely to their size, their wireless connections and hence their mobility. Battery power, however, can be an inhibiting factor that determines the range of viable computational and communication activities that can be supported [1].

Mobility constraints imply that system design should seek to optimise the use of techniques and resources that can prolong battery life. Computationally intensive operations such as key generation in asymmetric encryption may pose a heavy burden on mobile devices. Moreover, the availability of heterogeneous mobile systems and their limited resources may require the adoption of lightweight and adaptive schemes and hence may favour platform-independent schemes.

Within this context the P2P model is seen as ideal for providing order and structure on mobile interactions and for exploiting the potential of mobile devices. A more disciplined approach to information management is complemented by the flexibility of a personal device. The affinity between mobility and P2P modes of interaction is further reinforced by the dynamic and heterogeneous domains they support. There are however legitimate concerns over reliability and security in P2P systems. These issues tend to be exacerbated in a mobile context.

Amongst the key factors that are driving the research in P2P systems, distributed file management has provided the rationale for a variety of applications. This particular aspect of file management displays a natural affinity with the distributed nature of P2P systems. Many of the schemes which were implemented for wired and fixed networks are gradually being migrated to mobile systems.

This work is aimed at deploying a file dispersal service in a mobile P2P environment. The contribution of this paper is two-fold. Firstly, it identifies the main characteristics of P2P systems and mobile devices and proposes an architecture that achieves synergy between them. Secondly, it presents file dispersal as an application in a mobile P2P network where issues of reliability, security and efficiency are addressed.

The remainder of the paper is structured as follows. Section 2 defines the research context. Section 3 identifies system requirements and gives an outline of the mobile P2P system architecture. Section 4 presents some experimental results. Section 5 offers some comparative evaluation and pointers for further work, and Section 6 concludes the paper.

2. Research context

The scope of this research is defined by two main issues. Firstly, the identification and potential integration of the salient features of mobile devices and P2P systems and, secondly, the investigation of the main modes of deploying a scheme for file dispersal in a distributed environment. This also subsumes secure communication.

2.1 Mobile Systems

Despite their storage and computational limitations mobile devices have been the subject of intense research in distributed systems. The scope of application domains where their versatility is considered an asset ranges from P2P systems [2] to Cloud computing [3]. Most of the research efforts have been guided by an overriding concern over power management and the need to satisfy other requirements, including Quality of Service (QoS). In some research programmes this was translated into the selection of efficient methods and techniques. This approach has the advantage of promoting energy awareness and encouraging the selection of lightweight algorithms.

2.2 P2P Systems

In contrast to client-server models where connectivity between clients and servers may be intermittent, the symmetric role that nodes play in P2P systems necessitates their continuous operation and availability. In addition, the heterogeneity of P2P systems and the absence of a centralised authority give rise to information management as well as security and reliability concerns [4, 5].

The architecture of P2P systems may conform to two models: unstructured and structured. In unstructured systems such as Gnutella, peers establish random connections, and lookup requests are explicitly forwarded by intermediate peers until the search is exhausted. In structured P2P systems, nodes are organised into a network overlay. These systems are often implemented by a Distributed Hash Table (DHT) algorithm, as in Chord [6], Pastry [7] and Kademlia [8]. Consistent hashing is used in DHTs to partition a key space among a set of peers. Files are identified by a combination of the hash value of their name and the hash value of their content. A node is assigned a hash value as node ID and is responsible for the management of a range of key values through a routing table. DHT-based systems are scalable, robust and the DHT routing protocol guarantees the convergence of the search in an efficient manner. Their capacity for self-organisation enhances their resilience.

The secure transmission of data can be achieved by symmetric or asymmetric encryption schemes. Although symmetric schemes tend to be faster and less computationally demanding than asymmetric schemes, they have to explicitly address the issue of key distribution. A hybrid solution which combines symmetric and asymmetric encryption, can offer an acceptable compromise by allowing the exchange of symmetric session keys using asymmetric algorithms.

2.3 File dispersal

One area where the convergence of mobile devices and P2P systems can be beneficial is in the use of distributed mobile storage for files. File dispersal for example can be used to conceal files that hold unstructured data and where search is infrequent. Confidential information such as memos and reports fall under this category. This type of file management demands an acceptable level of reliability and is closely associated with data redundancy.

Two main forms of replication have been identified: monolithic and fragmented. In the monolithic method the entire file is replicated into a number of copies and dispatched to collaborative nodes for storage. Subsequent retrieval of a replica of the file requires access to only one node. With fragmentation a file is split into a number of fragments, which are distributed randomly to different nodes. The original file can only be reconstructed by retrieving all the fragments of the file. Different levels of replication can be set by replicating and distributing multiple copies of files or fragments. Although replication schemes can be easily implemented, the potential vulnerability of peers may require a high level of replication.

Erasure coding is another form of data redundancy [9, 10]. Erasure coding is implemented by an information dispersal algorithm (IDA), where the initial s fragments of a file are encoded into $s+r$ fragments and subsequently dispersed over the network. Since only a subset s of the fragments is needed to reconstruct the original file, erasure coding has been considered as more resilient to node failure and has been implemented in many systems.

3. Design and architecture

The enhancement of P2P modes of interaction with mobility may give rise to conflicting requirements implied by two overriding concerns: the intrinsic volatility and vulnerability of P2P networks, and the inherent limitations of mobile devices. These concerns are addressed by ensuring the security and reliability of storage and interactions, and by implementing efficient solutions. Core requirements for the design and implementation of the system are further augmented by provisions for flexibility and configurability. These were translated into two main design decisions:

1. The adoption of hybrid network where mobile devices are supported by a base station to ensure resilience and persistence. The bootstrap process is supported by the fixed base station.
2. The implementation of a DHT-based P2P network. DHT schemes support the mapping of fragments to nodes and the quick convergence in the search for fragments and peers. Structured overlays impose a structure on the topology of the network and on the assignment of data to nodes. These constraints are designed to improve the performance of data discovery and retrieval [11]. The main function of the system is to store and retrieve well-identified blocks of data. As

the discovery process is essentially one of retrieval a structured overlay is more suitable for the application.

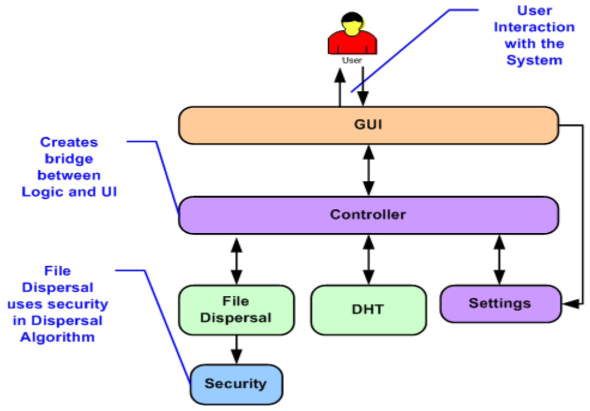

User Interaction with the System
User
GUI
Creates bridge between Logic and UI
Controller
File Dispersal uses security in Dispersal Algorithm
File Dispersal
DHT
Settings
Security

Figure 1. Architectural components

3.1 Architectural components

The system was designed as a layered architecture with file dispersal at the highest level, supported by the DHT Chord network; with communication forming the lowest layer. The different components of the system are presented in Figure 1 and detailed below.

Graphical User Interface (GUI)

GUI allows direct interaction between user and system. It has been designed within the constraints of screen area of mobile devices. Various interfaces were developed to provide status information and support for configuration.

Controller

The Controller is the main module that acts as a mediator between GUI and the system logic and data persistence. It integrates the different components of the system and performs the main tasks, from DHT instance creation and management of application level settings to file dispersal management.

File Dispersal

The file dispersal process can be realised either through replication or erasure coding. The file dispersal process involves a number of common steps:

1. File selection
2. File compression
3. File fragmentation
4. Fragment encoding (for erasure coding only)
5. Fragment encryption
6. Fragment distribution and
7. Metadata encryption and secure storage

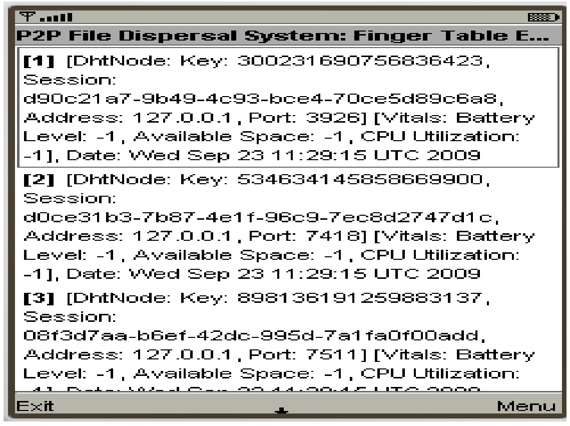
The fragmentation procedure generates a random fragment size within the limits set by the user. Every fragment is given an order number and its data saved in memory. Each fragment is encrypted by randomly applying one of several available symmetric encryption algorithms; the symmetric key is generated from the username and the password of the current user. DHT keys are generated by the application of the SHA-1 function to each fragment so as to identify the peer to which it will be dispatched and where it will be stored. The dispersal process is configurable and the user is able to select the type of file dispersal, replication or erasure coding, and also specify the level of replication. Different levels can be set for replication:

1. No replication (1 Key generated; original file only)
2. Low Replication (2 Keys; 2 copies of the file)
3. Medium Replication (3 Keys; 3 copies)
4. High Replication (4 keys; 4 copies)

P2P management

The Chord overlay network was selected for the DHT implementation thanks to its simplicity and scalability. It has the ability to guarantee efficient convergence in search queries, which is *O(logN)* where *N* is the number of nodes in the network. Its relative efficiency is also a valuable attribute in a mobile context. In a Chord network peers form a ring topology and every peer has a predecessor and a successor. The network can have a maximum of 2^m peers where *m* is the key size in bits; the peers are arranged in a circle with addresses from *0* to $2^m -1$. Every peer contains a routing table, the finger table, which has links to other peers in the network and whose size is equal to at most *m*. A modified version of the SHA-1 function was used for the generation of the keys. The key length was reduced to 64 bits and base 10 used to facilitate the efficient computation of the distance metric for the topology.

Chord lacks some important features such as the dynamic building of the routing table, a facility that has implications for communication overheads and resilience against DOS attacks. This is an intrinsic feature of Kademlia, which has been incorporated into this implementation of Chord. Instead of allocating all of the 2^m slots to the finger table at the outset, the slots are built dynamically in the form of a linked list when new peers are encountered (Figure 2). Entries to the Chord finger table are made on the basis of



```
[1] [DhtNode: Key: 3002316907568336423,
Session:
d90c21a7-9b49-4c93-bce4-70ce5d89c6a8,
Address: 127.0.0.1, Port: 3926] [Vitals: Battery
Level: -1, Available Space: -1, CPU Utilization:
-1], Date: Wed Sep 23 11:29:15 UTC 2009
[2] [DhtNode: Key: 5346341458586669900,
Session:
d0ce31b3-7b87-4e1f-96c9-7ec8d2747d1c,
Address: 127.0.0.1, Port: 7418] [Vitals: Battery
Level: -1, Available Space: -1, CPU Utilization:
-1], Date: Wed Sep 23 11:29:15 UTC 2009
[3] [DhtNode: Key: 8981361912598883137,
Session:
08f3d7aa-b6ef-42dc-995d-7a1fa0f00add,
Address: 127.0.0.1, Port: 7511] [Vitals: Battery
Level: -1, Available Space: -1, CPU Utilization:
```

Figure 2. Finger table

seniority. This tends to increase the number of known valid nodes in the network and fosters a more stable network. The system relies on a pinging mechanism to identify live peers and keep the routing information up to date. Peers may not respond due to intermittent Internet connection, crash or battery failure. This feature is also useful in detecting duplications and thus reducing processing and storage overheads. The system is designed to enable mobile devices to communicate over the Internet, with TCP as the Transport level protocol. The ubiquity of TCP facilitates the integration of heterogeneous components.

Security

Different aspects of security are provided: authentication, secure communication, and the encryption of the data and metadata. A session-based mechanism forms the basis of the authentication process. It is enhanced by the automatic generation of public/private key pairs and by the registration of the public key with the server.

Data protection is assured by the provision of light versions of six symmetric algorithms so that they can work efficiently on a mobile platform. They include DES, DESEde (Triple DES), IDEA (International Data Encryption Algorithm), Twofish, Rijndael (also called AES) and its light version AESlight. By default, AES is used for symmetric encryption and RSA for asymmetric encryption. An additional guard involves ensuring the integrity of the data through MD5 message digests. A critical component of the security apparatus is the secure storage of metadata and its access. Two main data stores in J2ME are allocated exclusively to each user based on the keys of the nodes, which in turn depends on the unique name of the users. Moreover, three encryption versions of the system were developed: symmetric, asymmetric and hybrid.

3.2 System behaviour

The P2P system was designed and implemented as a DHT-based Chord overlay network. In Figure 3 a snapshot of a mobile network is presented after a few peers have successfully joined it. The system includes a bootstrap server and a database server with mobile nodes as peers. When a peer joins the network for the first time the application settings are loaded. Once a peer bootstraps itself with the server, it registers its public key. If a peer needs to communicate with another peer, it requests the public key of the other peer from the server. Only authenticated peers can gain access to public keys. The public key of the other peer is cached in the routing table and remains valid as long as the other peer is alive or the session is active. The key is also used to insert the node in the network so that the Chord network is built gradually.

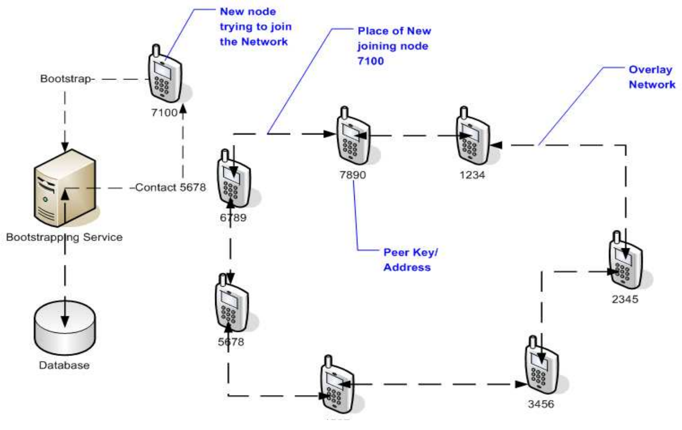


Figure 3. Network architecture

In the bootstrap process the server checks with the database server for the existence of the peer. If the peer is already registered the server creates a session for the peer and returns a session ID. The session ID is used for verification and for any communication with other peers in the network. Once the authentication is complete, the peer makes a request to the server for a bootstrapping node in the network. The server returns the address of a randomly selected node, which is then contacted by the initiating peer and presented with its public key and a session ID. For the peers that have joined the network the communication between them takes place without the intervention of the bootstrap server. All subsequent communication is also recorded in the routing tables. Messages between peers are encrypted and transmitted with a message digest.

3.3 Implementation

In the development of the system, preference was given to open source software and compatibility with mobile environments. The system was implemented in Java which has a robust set of well-defined APIs specifically designed for mobile devices; it offers good support for emulators in Java ME. Although J2ME is suitable for mobile devices it lacks some of the functionality of the standard J2SE. A number of modules had to be ported from J2SE software.

As the first point of contact for new peers with the network, the bootstrap service was designed to be responsive and flexible. It was implemented as a Web service and hosted on GlassFish. The Bouncy Castle Crypto API formed the basis for the security features of the system [12]. This API is also J2ME compatible and provides a rich set of cryptographic functions. File compression was realized with jazzlib [13], a light-weight Java API. It was converted and adapted into a J2ME light version.

The erasure coding function is based on the Java version written for the JigDFS distributed file system [14]. It is an implementation of the Cauchy Reed Solomon information dispersal algorithm, which was also converted into a J2ME version. The system was deployed on the mobile environment provided by the Sun Java Wireless toolkit 2.5.2 for CLDC. It offers extensive support for mobile emulators and for the creation of a realistic environment.

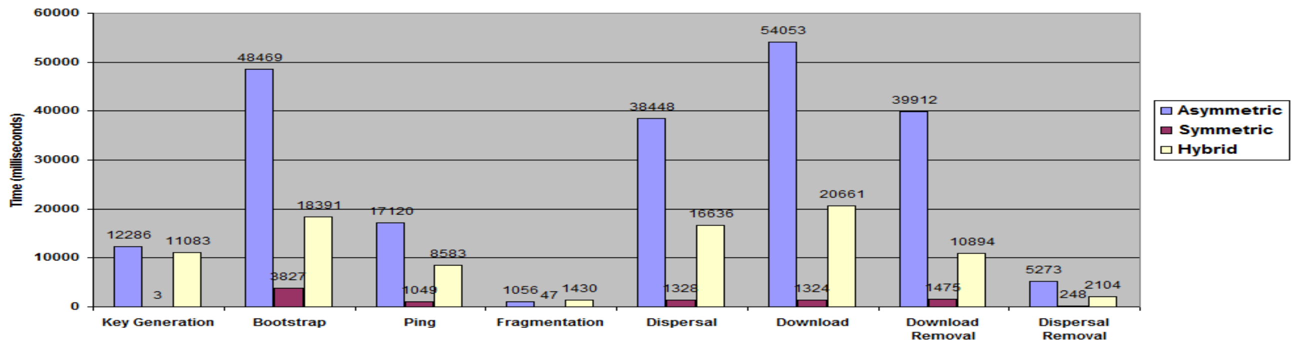


Figure 4. Functional performance

4. Experimental Results

A number of experiments were conducted in order to gain an insight into the efficiency and reliability of the system.

4.1 Efficiency

As encryption is a computationally intensive process, comparing the performances of the three different versions would demonstrate the efficiencies of the system and its use within a realistic context. The performance of the different versions of the system was measured in terms of the time taken from the initiation of the dispersal process up to its completion. The experiment was run with two peers for each version of the system.

In the graph in Figure 4 the significant difference between the performance of the symmetric and the asymmetric versions is evident, whilst the performance of the hybrid system falls between the two schemes. The key generation for the symmetric version is efficient on the mobile platform while in the asymmetric implementation it is relatively slow. As the hybrid system requires the generation of a key pair it takes a similar length of time as the asymmetric scheme. The fragmentation process in the hybrid system appears to be less efficient than in the asymmetric version; this difference in efficiency is due to the application of compression. Compression has the advantage of reducing the time taken for the fragmentation, the dispersal and the download. For the most important functions, such as bootstrapping, dispersal and download, the graph also indicates that the asymmetric version is inefficient.

These results show that a symmetric encryption implementation outperforms the other schemes and suggest that a pure asymmetric encryption scheme is not suitable for such a system. As the exchange of symmetric keys may be a source of vulnerability and can be awkward to perform the hybrid scheme appears therefore as a suitable compromise in such an environment. It combines the advantages of the other two systems, without their drawbacks. An improvement of the throughput of the system may be achieved by an optimisation of the file dispersal functions.

Attributes	Replication	Erasure
Original Fragment Count	**10**	**9**
Total Fragment Count	**30**	**27**
Average Fragment Size	**763**	**914**
Total Space Used	**22890**	**24678**
Keys per Fragment	**3**	**1**
Total Keys Generated	**10**	**27**
Fragments per Iteration	**3**	**1**
Total Iterations	**10**	**27**
Time taken (milliseconds)	**200**	**1236**

Figure 5. Erasure coding and replication

4.2 Replication and erasure coding

It has been established that erasure coding is more robust than pure replication [9]. The comparison between erasure coding and replication, in Figure 6, is concerned with the resource utilization in both implementations. In the proposed system Medium Replication is the equivalent of erasure coding. It generates three copies for each fragment; the number of encoded fragments created in erasure coding is also three times the original number of fragments. A file of 14846 bytes was compressed to 7630 bytes of binary data and then subjected to the IDA. Figure 5 lists the criteria over which the two methods were evaluated. As the erasure coding algorithm adds redundant information, the method consumes more space than the simple replication technique. Replication on the other hand generates fewer keys in total, involves less iteration over encryption, and is much more efficient than erasure coding. Overall, erasure coding used 7.8% more space, while its run-time was also at least five times higher than the simple replication. Erasure coding generates higher communication overheads.

4.3 Resilience

A higher level of replication can enhance the robustness of the system: the probability that all the fragments of a file can be recovered is very high, even when some of the peers are down or offline. An experiment was performed with five peers in order to evaluate the effectiveness of replication. To ensure consistency one specific peer was responsible for dispersing and downloading the file throughout the experiment

A file of size 14846 bytes was used with the default fragmentation settings. In Figure 6 the table displays the experimental results for the same file with different replication levels and for a varying number of offline peers. Peer selection was random so as to simulate the non-deterministic behaviour of the network peers. A *'recovered'* entry in the table indicates that the file was successfully reconstructed while a *'failed'* denotes that the file was not fully recovered. The table indicates that the highest number of file retrievals occurred when the highest replication level was set. The file was successfully downloaded even when all the peers were offline. This is due to the fact that there were only four other peers with the owner; for erasure coding there were enough replicated fragments that resided on the store of the owner. With no replication, even when one single peer was offline, the file could not be recovered in full. The peer, which was offline, was holding some of the fragments that were needed by the owner to rebuild the entire file. This experiment confirms that with a threshold of three, erasure coding is always successful in reconstructing the original file. The results underline the adequacy of replication in stable and reliable environments, and the advantage of erasure coding in unstable environments.

	Erasure coding	**High replication**	**Medium replication**	**Low replication**
0 offline	***recovered***	***recovered***	***recovered***	***recovered***
1 offline	***recovered***	***recovered***	***recovered***	***failed***
2 offline	***recovered***	***recovered***	***recovered***	***failed***
3 offline	***recovered***	***recovered***	***failed***	***failed***
4 offline	***recovered***	***failed***	***failed***	***failed***

Figure 6. System resilience

5. Discussion

The rationale for this research stems from the convergence of three types of constraint. Firstly, at the heart of the P2P perspective lie pointers for security and reliability. Secondly, a potentially unstable and unreliable environment requires flexibility and configuration in system deployment and thirdly, from a mobile perspective, an energy-aware design calls for efficient and appropriate solutions.

5.1 Security

Security procedures were woven into the fabric of the system from the outset. Security was catered for along two main threads: ensuring authentication and securing data while it resides on nodes and during its transfer.

Authentication

A session-based authentication scheme was deployed in order to provide a safe environment. Although this scheme could be subjected to attacks such as session hijacking [15], the impact of malicious attacks is mitigated by hybrid encryption and fragmentation. A hybrid approach to authentication enhances the role of the bootstrap server by augmenting its functionality with centralised certification. The choice of session-based authentication and hybrid encryption is an example of the compromises made between security and efficiency.

Data security

The issue of data security is partially addressed by the nature of the application itself and by the restriction that access to files is only-read; the update problem does not arise. Confidentiality is maintained since the dispersed fragments are compressed and encrypted. In some schemes fragmentation by itself is considered an effective means of securing data [16]. Confidentiality is also assured by the encryption of the metadata.

5.2 Reliability

The reliability of the system is the outcome of the reliability of the different levels of the architecture: at the application level, DHT level and P2P network level. This is discharged at the application level by the provision of two modes of file dispersal, and at the P2P level by a combination of authentication and encryption. The reliability of the system is enhanced by the inherent features of erasure coding and its suitability for unstable environments. It is also reinforced by the levels of replication afforded by the replication scheme.

Overlay Network architecture

The benefits that stem from the overlay network are due to the following features:

1. Decoupling of information from location,
2. Resilience of the system to high churn rate through Chord's capacity for self-management,
3. Reduction of DOS attacks through enhancement with Kademlia's routing tables, and
4. Handling of the heterogeneity of nodes through widely deployed and supported system software.

The provision of a bootstrap server enhances reliability and supports persistence. Its role can be expanded to include additional functions such as brokering and load balancing.

Flexibility

Flexibility and configuration represent another facet of the system. They help deal with the openness of P2P systems, their dynamic and ad hoc nature. The user is able to set various parameters and to select the appropriate method for performing file dispersal to suit environmental conditions. The flexibility of the system is also improved by the portability that follows the choice of J2ME as a platform for implementation. This covers ease of redeployment and resilience in heterogeneous environments.

5.3 Efficiency awareness

Effective power management in mobile devices is evidently critical to their viability. Gurun et al [17] considered energy consumption in a P2P structured system, and the main conclusions of their investigation are that the application layer management can help save energy, and that lightweight protocols can improve performance and reliability. The relevance of their approach at this level was confirmed by a study of file sharing by Nurminem et al [18]. In addition, one performance investigation on the deployment of DHT-based mobile P2P systems [19] indicates that one of the main sources of energy consumption is due to the processing of incoming messages.

The development of the proposed system is in line with the first conclusions. Core design decisions were aimed at minimising storage usage, computational load and therefore communication load on individual nodes. Relevant techniques include file compression, file fragmentation and flexible fragment size. A significant reduction in the size of the files and computational and storage costs was achieved by applying compression algorithms. For the implementation of SHA-1 and for the encryption algorithms, light versions such as AESlight were introduced in order to facilitate topology computations and improve efficiency. Furthermore, three different schemes were implemented with different efficiency levels; session management falls within efficiency considerations.

The experiments conducted on the emulator provide a sound basis for a comparative evaluation. These results are however relative and are recorded in a small simulated environment. The porting of the system and its deployment to a real and wider mobile environment is likely to have an adverse effect on memory capacity and on the overall performance [20]. It is also worth noting that in a highly mobile context the TCP protocol may not be suitable.

5.4 Related work

Attempts at deploying P2P systems over mobile environments have been marked by the introduction of various schemes. In Kato et al [21] a hybrid architecture is proposed which integrates mobile devices into a P2P system through the use of proxy nodes. The work is marked by the implementation of appropriate protocols. The development of Proem [22] was motivated by the need to create an environment which facilitates application development. JXTA is a framework which has successfully integrated mobile devices and P2P systems [23]. In contrast with earlier work where the focus was on protocol optimization, Wu [24] proposes a layered architecture for secure and trusted transactions in a mobile environment. The work proposed in this paper relates to the deployment of higher level protocols and applications.

File dispersal has been implemented in a number of systems. The Farsite system [25] is an example where reliability and high availability are addressed by replicating an entire encrypted file on different nodes. This method is easy to implement but the availability of the entire file on each node makes it vulnerable to malicious attacks. Furthermore, the transmission of multiple files can consume a lot of bandwidth. Among the systems that implement an information dispersal algorithm Cleversafe [16] relies on the size of the fragment as a security barrier and does not encrypt any data. The emphasis is on data reliability through mere replication. A more focused approach to data reliability and distribution is adopted in OceanStore [26] where fault-tolerance is ensured by erasure coding. In JigDFS [14] in the implementation of erasure coding, the process is further refined by the recursive encryption of the file fragments. IgorFS [27] is a distributed file system based on a Chord-like network. It is similar to the fragmented replication but the key generation involves a higher level of iteration on the hashing and encryption functions.

With few exceptions, most of these systems ensure fault tolerance through an IDA. They were developed however for fixed and wired systems only. In contrast, the proposed system is a P2P system for a mobile context. In addition to the provision of two modes of file dispersal, the system is also supported by a DHT-based overlay network.

5.5 Further work

The reliability of the system depends on the security procedures of the system only, and assumes the existence of trusted peers. This system is vulnerable to malicious peers and to denial of service (DOS). This situation can be remedied by the enhancement of the authentication process by a certificate-based scheme, and by incorporating a trust and reputation layer [5]. This additional layer can improve the quality of file dispersal and retrieval. The resulting system may be however very complex and incur prohibitive storage and communication overheads.

Although a trust layer can improve the reliability of the transactions it does not address the issue of the high churn rate that occurs in P2P systems. A defective node can destabilise the network and incur considerable overheads. The viability of the file dispersal application is dependent on a relatively stable environment. In the current implementation the metadata is stored on the node that initiated the dispersal and the system relies on J2ME libraries for encryption and confidentiality. A higher level of security can be achieved by dispersing the metadata itself. This would be particularly relevant to erasure coding because of its fault-tolerance. Further work should also aim at optimising some of the functional components of the system. Another important issue for consideration involves the porting of the system to a real and larger mobile P2P system and an evaluation of its behaviour. A more fundamental matter concerns the constraints that DHTs might impose on the viability of the distributed application. The tight coupling between IP address and node identifier may be an obstacle to the movement of mobile devices across multiple administrative domains; a device may be assigned different IP addresses. Some form of residual dependency and forwarding may have to be considered.

6. Conclusion

This paper has presented a secure, reliable and efficient system for file dispersal on a mobile DHT-based P2P network. Various methods, technologies and lightweight algorithms were integrated in the design and the implementation in order to accommodate the characteristics of the mobile systems and P2P networks. The system manages to reconcile various requirements and to strike a balance between reliability and security without sacrificing efficiency. The availability of a number of encryption schemes and of two modes of file dispersal is a key factor in the flexibility and configurability of the system. It enhances its resilience and its suitability to different environments.

7. References

[1] Walkerdine J. and Lock S.; Towards Secure Mobile P2P Systems, *Second Int. Conf. on Internet and Web Applications and Services (ICIW '07),* 2007.

[2] Qureshi B., Min G., and Kouvatsos D.D. M-Trust: A Trust Management Scheme for Mobile P2P Networks. *The IEEE/IFIP 8th Int. Conf. on Embedded and Ubiquitous Computing (EUC 2010),* 2010, pp476-483.

[3] Klein A., Mannweiler C.,Schneider J. and Schotten D. Access Schemes for Mobile Cloud Computing. *The 11th Int. Conf. on Mobile Data Management*, 2010, pp387-392.

[4] Wang Y. and Vassileva J., Trust and Reputation in Peer-to-Peer Networks, *The 3rd IEEE Int. Conf. on Peer-to-Peer Computing*, Sweden, 2003, pp150-157.

[5] Anane R., Marrocco S. and Bordbar B. Trusted P2P Group Interaction. *The 2nd Int. Conf. on Computer Science and its Applications (CSA 2009),* IEEE Publication, Korea, December 2009, pp1-8.

[6] Stoica I., Morris R., Karger D., Kaashoek M.F. and Balakrishnan H., “Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications”, MIT Laboratory for Computer Science, SIGCOMM’01, August 2001, San Diego, California, USA. *http://pdos.lcs.mit.edu/chord/*

[7] Rowstron A. and Druschel P. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems, T*he 18th IFIP/ACM Int. Conf. on Distributed Systems Platforms (Middleware 2001)*. Germany, November 2001.

[8] XLattice, Kademlia: A Design Specification, *http://xlattice.sourceforge.net/components/protocol/kademlia/specs.html.*

[9] Weatherspoon H., Kubiatowicz J., Erasure Coding Vs. Replication: A Quantitative Comparison, *1st Int. Workshop on Peer-to-Peer Systems,* March 2002. p.328-338

[10] Rodrigues R. and Liskov B. High Availability in DHTs: Erasure Coding vs. Replication, *4th Int. Workshop Peer-to-Peer Systems (IPTPS 2005),* New York, February 2005, pp226-239.

[11] Castro M., Costa M. and Rowstron A. Performance and Dependability of structured peer-to-peer overlays, *Int. Conf. on Dependable Systems and Networks (*DSN-2004), Florence, Italy, June 2004

[12] Bouncy Castle Crypto APIs, The Legion of the Bouncy Castle, *http://www.bouncycastle.org/*

[13] A pure java implementation of the java.util.zip library, *http://jazzlib.sourceforge.net/*.

[14] Bian J. and Seker R., “JigDFS: A Secure Distributed File System”, *IEEE Symp.* on *Computational Intelligence in Cyber Security CICS '09.* USA, March 2009, pp76-82.

[15]] Xiaobo L., Sikdar, B. A mechanism for detecting session hijacks in wireless networks., *IEEE Transactions on Wireless Communications, Volume: 9, Issue: 4*, 2010, pp1380 – 1389

[16] Advancing dispersed storage. *http://www.cleversafe.org/.*

[17] Gurun S., Nagpurkar P. and Zhao B. Energy Consumption and Conservation in Mobile Peer-to-Peer Systems, Proceedings of *International Workshop on Decentralized Resource Sharing in Mobile Computing and Networking (ACM Mobishare),* USA, 2006, pp18-23

[18] Nurminen J. K and Nöyränen J., "Energy-Consumption in Mobile Peer-to-Peer - Quantitative Results from File Sharing", *5th IEEE Consumer Communications & Networking Conf. (CCNC),* USA, January 2008, pp730-733.

[19] Kelényi I and Nurminen J.K: Energy Aspects of Peer Cooperation . *43th IEEE Int. Conf. on Communications (ICC 2008),* Beijing, 2008.

[20] Wang, A.I., Bjornsgard, T. and Saxlund, K. Peer2Me - rapid application framework for mobile peer-to-peer applications, *Int. Symp. on Collaborative Technologies and Systems, 2007 (CTS 2007),* USA, 2007 , pp379-388.

[21] Kato T., Ishikawa N., Sumino H., Hjelm J., Yu H. and Murakami S., A platform and Applications for Mobile Peer-to-Peer Communications, *5th IEEE Consumer Communications and Networking Conf. (CCNC 2008),* January 2008, pp1176-1180

[22] Kortuem G., Schneider J., Preuitt D., Thompson T.G.C, Fickas S. and Segall Z. When Peer-to-Peer comes Face-to-Face: Collaborative Peer-to-Peer Computing in Mobile Ad hoc Networks, *First Int. Conf. on Peer-to-Peer Computing (P2P'01),* 2001.

[23] Maibaum N. and Mundt T., JXTA: A Technology Facilitating Mobile Peer-To-Peer Networks, *IEEE Mobility and Wireless Access Workshop (MOBIWAC 2002*), pp7-13

[24] Wu X., Trusted Architecture for Mobile P2P Systems, *6th Int. Conf. on Wireless Communications Networking and Mobile Computing (WiCOM 2010),* China, Sep 2010, pp1-4.

[25] Farsite. *http://research.microsoft.com/en-us/projects/farsite/*

[26] The OceanStore Project. *http://oceanstore.cs.berkeley.edu/*

[27] Amann B., Elser B., Houri Y. and Fuhrmann. IgorFs: A Distributed P2P File System. *2008 Eighth Int. Conf. on P2P Computing*, Germany, 2008, pp77-78