

Community Trust DRM in Peer-to-peer Networks

Madjid Merabti, David Llewellyn-Jones

School of Computing and Mathematical Sciences, Liverpool John Moores University
James Parsons Building, Byrom Street, Liverpool, L3 3AF, UK

{M.Merabti, D.Llewellyn-Jones}@ljmu.ac.uk

Abstract—The success of the Ubiquitous Computing paradigm relies on fluid movement of data throughout the networked environment. This requirement is often seen as being in conflict with the need to be able to restrict the movement of certain rights protected data. In this paper we build on our previous work on community based DRM to show how it can be applied in an existing peer-to-peer filesharing network. By extending the Gnutella peer-to-peer filesharing protocol we show how a realistic implementation of a community-based trust mechanism can be achieved. Our mechanism models the peer-to-peer network as a cellular automaton, with each node maintaining a state, shared between neighbouring nodes and acted on by a simple transition function. Through the application of this function a trust level is automatically maintained for measuring the likelihood that a neighbour will transmit illegitimate (rights protected) data around the network. Consequently, we are able to use this to isolate rogue nodes and reduce sharing of copyrighted data illegitimately. We argue that through the creation of a realistic implementation of the technique in a peer-to-peer filesharing network, the viability of the scheme for use in a Ubiquitous Computing environment can be assessed.

Key words: Digital Rights Management, Ubiquitous Computing, Peer-to-peer filesharing networks, Trust.

1. Introduction

As computers devices become smaller, more mobile and more ubiquitous, we are seeing a fundamental shift in the way they are utilised. Not only have people become more comfortable and fluent with their use, but more fundamentally the focus of users' requirements has shifted from hardware to data. Easy, ubiquitous and fluid access to data and content has become far more important, as witnessed by the massive growth in music sales purchased through online providers [1].

In assessing the importance of this shift, we should consider that it has been driven almost entirely by users finding solutions to their own requirements. In contrast, traditional content suppliers have seemingly only grudgingly accepted the need for users to have greater networked availability of content. We might conclude that a trend driven so clearly by user requirements is likely to continue further.

However, it also raises the question as to why companies would be so reticent to capitalise on such an inexorable process. The answer, of course, is that there exists a dangerous conflict between functionality that allows a user to access and share his or her own data, and the rights of content producers not to have their data distributed if they so choose.

1.1. Digital Rights Management

Work in Digital Rights Management (DRM) has therefore been aimed at allowing the distribution of data to be controlled more effectively with the aim of protecting the rights of content producers and owners.

Various highly ingenious methods have been proposed that range from fingerprinting [2], watermarking [3], data encryption schemes [4] and trusted computing [5]. Of these, there is a distinction to be made between technology that allows the protection and detection of rights-enabled content, and technology that is intended to facilitate the management of such protected content.

Whilst detection of rights information via embedded data (through any of various methods such as metadata, watermarking *etc.*) and prevention of removal is a necessary prerequisite for most management schemes, it is not the focus of our work presented here. Instead, we are interested in how, assuming the existence of such data, it is possible to manage content distribution in a manner fair to both content users and content producers. In addition, we focus particularly on techniques appropriate for the mobile, dynamic and networked computing environments envisaged of the future [6]. Nonetheless, we do acknowledge that detection of rights information may not be a guaranteed process. This distinguishes our solution from other rights management schemes – such as those built on the Trusted Computing Platform (TCP) [5] – where in general it follows that a single breach of a system results in an exploitable vulnerability that can be universalised. As we hope to demonstrate, this need not necessarily be the case.

We achieve the unique features of our system by capitalising on the same features that make rights management so difficult: the fluid movement of data.

In particular, rather than assuming that all protection and security relating to DRM must reside on every single

machine in order to prevent every instance of a violation where it occurs, we instead turn to the wider community to apply such mechanisms, achieved through it an automated reputation and trust network.

1.2. Peer-to-peer Networks

The trust process we have developed has been described as it applies to Ubiquitous Computing in earlier work [7], where simulation results for such a Ubiquitous Computing network were presented. In the present paper we extend this using a trust protocol overlaid onto the Gnutella network.

Peer-to-peer filesharing networks present a relevant test space for various reasons. The peer-to-peer structure resembles what we believe to be the likely structure of future Ubiquitous Computing, where *ad-hoc* networks of devices are formed as part of the larger Ubiquitous network. Moreover, their aim – the sharing of data between devices – corresponds with a major benefits envisaged of Ubiquitous Computing environments, where data-centric interaction is likely to be key. Whilst at present, ubiquitous devices such as mobile phones – seen as being promising drivers of the Ubiquitous Computing trend – do not offer these *ad-hoc* and liberal data-sharing architectures, we believe this is likely to change as such devices increase in capability. For example, the first mobile phone Internet server was recently released by Nokia for use on Symbian platform phones [8], and the trend suggests we are likely to see more emphasis on mobile-to-mobile data sharing in the future.

We will examine the structures of peer-to-peer networks in more detail later. However, before providing details of how community trust can be applied in such environments, we must first consider how community trust mechanisms can work in general.

In the next section we will describe the community trust process in more detail, providing details of a trust overlay based on cellular automata techniques. We will then go on in Section 3 to describe our implementation based on the Gnutella network. Finally we will present conclusions and a discussion of future work.

2. Community Trust

In contrast to traditional DRM enforcement mechanisms (such as TCP) that rely on protection being active on the device where contravention occurs, we have aimed to develop a system that relies on community enforcement of digital rights. This is achieved through devices collaborating and periodically checking the content distributed by their neighbours.

An overlay trust network is used and at each node in the network a minimal state is maintained that is used to measure the legitimacy and trust at each node. This state is updated based on a simple algorithm that allows the entire network to act as a cellular automaton. In this way, we are able to capitalise on the properties of cellular automata that

allow local effects to produce global consequences. We balance the need for users to occasionally distribute data (for example under the terms of ‘fair use’) whilst penalising those who consistently flout copyright through the wholesale distribution of copyrighted data.

2.1. Using Cellular Automata

For our cellular automaton trust scheme, each node in the network is taken to be a cell of the cellular automaton. Cells of a cellular automaton are traditionally organised into a (rectilinear) grid, with each cell bordered by eight neighbours. We translate this into a more complex network by having nodes of the network represent cells of the cellular automaton, with neighbouring nodes being adjacent cells of the cellular automaton. As a result, we are unable to project this onto a regular grid, since we generally allow for nodes to have variable numbers of neighbours. Although this does affect the details of the scheme and we must be careful to weight the effect that neighbours have on a node accordingly, simulations show that the overall effect for our chosen transition function is not dissimilar to that seen with a regular cellular automaton.

For our trust scheme the state s_n of each cell n at time t is represented by three real values as follows.

$$s_n(t) = \begin{pmatrix} l_n(t) \\ v_n(t) \\ \tau_n(t) \end{pmatrix} \quad (1)$$

The values l_n , v_n , and τ_n represent the *legitimacy*, *velocity* (rate of change of legitimacy) and *trust* at each node. Intuitively, we can take the legitimacy to mean the belief a node has that it is likely to distribute only legitimate data from its neighbours (a higher value representing a greater confidence that only legitimate data will be used), whilst the trust represents the trust a node has in its neighbours. The two values are subtly different, in that the trust value will affect the legitimacy, but a node can maintain high legitimacy even with low trust by increasing the amount of checking that it undertakes (to be discussed presently). Note that the trust value relates to the neighbouring nodes, rather than the node that maintains the state itself.

The change of these values is governed by the transition function applied to the state of a node, as well as the state of the neighbouring nodes. This is defined mathematically so that, if a node detects illegitimate data transmitted by a neighbouring node, the trust level τ_n of the node is decreased. This ultimately affects the legitimacy l_n of a node. To see how this works, it can be illuminating to imagine the nodes projected into three-space with l_n representing the height of a node (see Figure 1, where a rectilinear network is depicted for clarity).

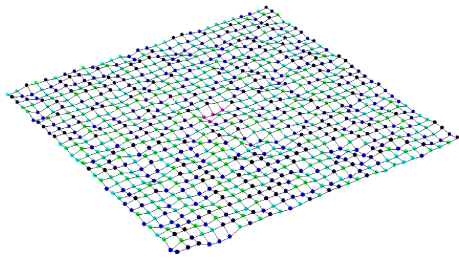


Figure 1: Representing network legitimacy in three-space

The trust level of the node acts as a vertical force causing changes to the (vertical) velocity v_n of the node. However, the velocity is also affected by other forces, notably the links between each node and its immediate neighbours. These links act as elasticated chords, so that a node with low legitimacy surrounded by higher nodes will be pulled upwards and vice versa.

The mathematical details of the transition function used are not provided here. Instead we refer the interested reader to our earlier work [7].

One final crucial mechanism relates the legitimacy of a node to the frequency of checking performed by a node. When the legitimacy falls below a certain threshold (which we set to be zero, but the actual value is somewhat arbitrary) a check of the data transmitted by neighbouring nodes becomes mandatory. A natural downward force is applied constantly to each node, causing it to drift downwards. However, each data check for which the data is found to be legitimate results in a counteractive upward force being applied to the node. Thus under normal circumstances, each node gently oscillates up and down, resulting in periodic checks of the data. The frequency of checks is dictated by the constants used for example to set the strength of each of the forces.

These constants should be set system-wide and are established through experiment. In the event that illegitimate data is detected, the resulting trust decrease causes an additional downward force to be applied to the node. Consequently the frequency of checking increases. Moreover, the greater the number of nodes detecting the transmission of illegitimate data, the lower the trust and legitimacy of the surrounding nodes becomes, and again this has an effect on checking frequency.

Due to the unpredictable nature that results from the transition function applied to multiple nodes, there is no way for a node to predict which pieces of data will be checked by neighbouring nodes without resorting to collusion.

Although we have not presented specific details, the above provides a more intuitive description of the process. As we shall see, the overall effect is that rogue nodes distributing copyrighted content are able to be isolated in the network, so that action can be taken against them.

2.2. Isolating the Movement of Illegitimate Data

As can be seen in Figure 2, nodes distributing illegitimate data can be established through data checking, and cause the surface of the network to be pulled downwards.

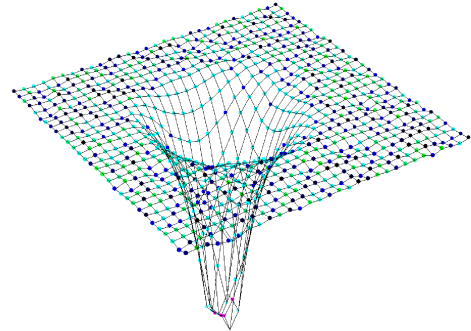


Figure 2: A rogue node isolated in the network

It is not our intention to consider what consequences such detected rogue nodes should suffer, however, the technique shows clearly how nodes can be isolated and thereby identified. Nodes that inadvertently transmit data illegitimately but which stop once discovered are not affected severely. However, nodes consistently distributing data illegitimately could have bandwidth throttling applied, be removed from the network completely, or have other sanctions imposed. The design of the transition function automatically accounts for the need for multiple nodes to discover the problem, so as to maintain some level of robustness against false accusations.

3. Implementation in the Gnutella Network

We have already undertaken extensive testing of the algorithms described above in simulated networks. In the forthcoming sections we extend our design to allow its use in peer-to-peer filesharing networks. Such networks can be used to demonstrate the practical application of the algorithm in real network scenarios.

Since the popular rise of Napster as a filesharing network around the turn of the millennium, a variety of alternative networks have been implemented, such as FastTrack (KaZaA), DirectConnect, eDonkey, AudioGalaxy, CAN, CHORD, Kad and Gnutella. Each has distinct architecture and properties [9]. For example, AudioGalaxy constitutes a mediated architecture, with a centralised control mechanism. CAN, CHORD and Kad are structured networks, where peer links are established based on a structured routing and resource discovery algorithm. For our purposes, we required a pure, unmediated and unstructured peer-to-peer architecture in order to mimic the unstructured peer-to-peer nature of *ad-hoc* ubiquitous computing networks. As an open architecture able to satisfy these requirements, we have therefore opted to consider the Gnutella network. Although Gnutella does have the

capability to provide a hybrid architecture that includes ultrapeers, we use only the pure peer functionality.

3.1. Gnutella Trust Protocol

Although the Gnutella protocol specifically allows for certain types of extension, we have chosen not to create a system that is necessarily compatible with the existing Gnutella network. Instead, we extended the existing Mutella servant for restricted testing in an isolated environment.

The standard message types used by Gnutella peers for communication are as shown in Table 1 [10].

Table 1: Gnutella messages

Type	Message	Description	Payload
0x00	Ping	Request for host to announce itself.	Does not generally contain a payload.
0x01	Pong	Response to a Ping.	Information about responding host.
0x02	Bye	Optionally used to inform a servant a host is connected to that it is closing the connection.	Error code and description string.
0x40	Push	Request a file to be pushed from a servant to a host.	Information about servant being requested to perform push, file to be pushed and destination host.
0x80	Query	Used to search the network.	Minimum speed and search criteria.
0x81	Query Hit	Response to a Query.	Hit results for the query being responded to.

We have extended this with a further message relating to the trust architecture as follows.

Type	Message	Description	Payload
0x03	State	Cell state	Legitimacy value l_n of the sending node.

This *State* message is used solely for the purpose of transmitting the current node state value to its neighbouring nodes. This allows the cellular automaton transition function to be applied to a node's current state along with the state of its neighbours. The transition function utilises only the legitimacy values l_n of neighbouring nodes (velocity and trust values are not needed in order to calculate the effective force exerted on surrounding nodes), hence it is only necessary to send a single value as payload. A single-precision float variable is adequate for this task. The complete state message is therefore structured as shown in Table 2 below.

Table 2: The State message format

Bytes	Value	Description
0-15	Unique value	Message ID
16	0x03	Payload type
17	0x01	Time to live (set to 1, since State messages should not be propagated).
18	0x00	Hops
19-22	0x04	Payload length
23-26	Dependent on node state	Payload: legitimacy value stored as 4-byte IEEE 754 standard single-precision binary floating point value.

A possible alternative implementation would involve the piggy-backing of legitimacy state onto Ping messages using the standard extension block. However, we deemed this to be suboptimal, since unlike Ping messages, State messages require no response. The use of a separate message therefore results in a reduction of overall bandwidth usage.

The State message is sent periodically to all of a peer's neighbours, based on a predetermined and network-wide frequency. Receiving servants cache the values until they are due to update their own state, a process that is undertaken with the same frequency as State message transmission. Caching is used to avoid the need for synchronisation, and once used for a state update, the cached legitimacy values of the neighbouring nodes are discarded. As a result, the dynamic structure of a peer-to-peer network can be automatically accommodated.

At present the integration of the trust mechanism into the Mutella servant is not fully completed and remains as ongoing work. However, having a prototype servant is only part of the process. A considerable challenge involves the deployment of a client in a way that allows meaningful observations to be made.

3.2. Evaluation Methodology

To deploy a special purpose Gnutella based network on an Internet-wide scale with multiple users is unviable. Instead, we can harness the known properties of existing Gnutella networks in order to validate generated simulation data.

Initially therefore, we simply run our extended Gnutella client on a restricted network. The Gnutella protocol allows for several servants to cohabit a single machine, and we can establish a restricted network of around a hundred extended Gnutella servants. The Mutella servant is altered to periodically download random files from other servants at uneven intervals. By sharing a mixture of both legitimate and restricted files within the network, we can record the state of each servant over time and the movement of content between servants.

The data thus obtained can be used to tailor our existing simulation based on cellular automata techniques. Our simulation is able to create a large (hundreds of thousands

of nodes) network satisfying predetermined small world and power law distribution properties using the Klemm-Eguiluz network generation technique. Studies of the pure Gnutella peer-to-peer network have established that they also satisfy both small world properties (high clustering coefficient and equivalent characteristic path length to the random graph) and a power law distribution. Javanovic, Annexstein and Berman [11] took measurements in late 2000 that established an approximate clustering coefficient $C \approx 0.0457312$ and characteristic path length $L \approx 3.61744$. The power law exponent was established to be $\alpha = -1.4$. These parameters can therefore be mimicked and through comparison of the experimental data generated using our extended Mutella servant with data generated using the simulated network, we can extend the simulation to allow larger networks to be explored in greater detail.

4. Conclusion

Considerable work has been undertaken and promising results established for the prospects of DRM through the application of community trust in Ubiquitous Computing networks in our earlier work. We have extended this so as to allow for its use in distributed peer-to-peer filesharing networks. In this paper we presented an extension of our work and the Gnutella protocol that allows for the trust mechanism to be overlaid onto an existing Gnutella peer-to-peer network using the Mutella client.

There clearly remains considerable work to be undertaken in the area and many questions that remain to be answered. For future work, we intend to provide a complete implementation of the Mutella servant incorporating the extensions to the Gnutella protocol. Although there is potential to allow such a client to interoperate with existing Gnutella servants through the extension capabilities that the protocol allows, we instead use a special purpose network that will allow us to undertake a more detailed analysis in an isolated environment.

Ultimately, data obtained using the extended servant used over a small network can be correlated with simulation data. In this way, the simulation may be extended in an effective way to allow larger scale and more accurate simulations in order to ascertain the effectiveness of the community trust based scheme in a realistic way.

DRM remains a hugely important area of research. Existing file sharing networks are stigmatised through their use as a means to share copyrighted data, but nonetheless have important legitimate uses as well. Moreover, the fluid and dynamic movement of content through a networked system remains a prerequisite for the deployment and acceptance of any true Ubiquitous Computing environment. The development of satisfactory Digital Rights Management technology that can allow for such fluid movement, whilst at the same time ensuring that rights holders are adequately protected, is therefore of paramount

importance to the future direction that computer development will take. We believe that a balance can be achieved between these two apparently conflicting goals. Moreover, our work has shown that the use of community based DRM enforcement remains a promising technique for the achievement of this goal. This realistic demonstration shows the viability of our proposed method.

References

- [1] "IFPI Digital Music Report," IFPI 19th January, 2006.
- [2] E. Allamanche, J. Herre, O. Hellmuth, B. Fröba, T. Kastner, and M. Cremer, "Content based identification of audio material using MPEG-7 low level description," in Proceedings of the International Symposium of Music Information Retrieval, Bloomington, Indiana, USA, 15-17 October 2001.
- [3] F. A. P. Petitcolas and R. J. Anderson, "Evaluation of copyright marking systems," in International Conference on Multimedia Computing and Systems, Florence, Italy, 1999.
- [4] Y. Jeong, K. Yoon, and J. Ryou, "A trusted key management scheme for digital rights management," *ETRI Journal*, vol. 27(1), pp. 114-17, February 2005.
- [5] S. Pearson, B. Balacheff, L. Chen, D. Plaquin, and G. Proudler, *Trusted Computing Platforms: TCPA Technology In Context*. Prentice Hall PTR, 2003.
- [6] G. Banavar and A. Bernstein, "Software infrastructure and design challenges for ubiquitous computing applications," *Communications of the ACM*, vol. 45(12), pp. 92-6, December 2002.
- [7] M. Merabti and D. Llewellyn-Jones, "Digital Rights Management in Ubiquitous Computing," *IEEE Multimedia*, vol. 13(2), pp. 32-42, April-June 2006.
- [8] J. Wikman and F. D. Rácz, "Mobile Web Server," 2006: Nokia Research Center, <http://research.nokia.com/research/projects/mobile-web-server/index.html>.
- [9] P. Backx, T. Wauters, B. Dhoedt, and P. Demeester, "A comparison of peer-to-peer architectures," in EURESCOM Summit 2002, Heidelberg, Germany, 21-24 October 2002.
- [10] "Gnutella Protocol Development," 2006: Gnutella Developers, <http://www.the-gdf.org/>.
- [11] M. Jovanovic, F. Annexstein, and K. Berman, "Scalability Issues in Large Peer-to-peer Networks - a Case Study of Gnutella," University of Cincinnati January 2001.