Securing Peer-to-Peer Networks Using Trusted Computing

Shane Balfe, Amit D. Lakhani and Kenneth G. Paterson,
Information Security Group,
Royal Holloway, University of London,
Egham, Surrey, TW20 0EX,
United Kingdom.

January 31, 2005

Keywords: TCPA, TCG, trusted computing, P2P, peer-to-peer, pseudonyms

1 Introduction

It seems likely that TCG-compliant computing platforms will become widespread over the next few years. Once one accepts that the Trusted Computing paradigm offers an interesting and powerful set of security features, the natural question arises: for what purposes can this technology be exploited? In this chapter, we examine the application of Trusted Computing to securing Peer-to-Peer (P2P) networks.

The concept of P2P networking covers a diverse set of network types, supporting a wide variety of applications. The common feature shared by almost all P2P networks is the lack of any centralised control. In this respect, P2P networks are the antithesis of the traditional client-server model. They have most famously become popular in the form of P2P file-sharing networks, providing a means of distributing (often copyrighted) material such as music and video. Commercially-oriented P2P networks are now coming to the fore. Useful introductions to P2P networking can be found in [23, 25].

Aside from availability, security issues for P2P networks have not yet been widely addressed. A major conflict arises from the perceived requirement to provide anonymity for users of P2P networks and an increasing need to provide robust access control, data integrity, confidentiality and accountability services. These services are increasingly important as industry moves towards using P2P technology in applications and as P2P e-commerce emerges. The security situation for P2P networks is made worse because, by definition, they lack any centralised authority who can vouch for identities or security parameters. Without the foundation of stable, verifiable identities, it is difficult to build any of the desired security services. In particular, pseudospoofing attacks, in which malicious parties claim multiple identities and disrupt the operation of P2P networks, are difficult to prevent. We provide an overview of the main security issues for P2P networks in Section 2.

In this chapter, we demonstrate how features of the TCG specification [31, 34] can be employed to enhance the security of P2P networks. In particular, we show how the TCG protocols for Direct Anonymous Attestation (DAA) can be used to enforce the use of stable, platform-dependent pseudonyms and reduce pseudospoofing in P2P networks. Further, our use of DAA provides a means of building entity authentication and simple access control

mechanisms. Taking this a step further, we show how runs of the DAA protocol, providing authentication of pseudonyms, can be securely linked to the establishment of secure channels with known endpoints. Such channels allow the protection of data in transit in P2P networks. We show how the cryptographic mechanisms we provide can be integrated with standard SSL and IPSec protocols. An important feature of our work emerges here: while earlier authors [16, 20, 29] have posited the potential application of Trusted Computing in P2P networks, we do not halt at that point. Rather, we go as far as describing the specific TCG mechanisms and commands which enable us to establish security in P2P networks.

After establishing how the basic security features can be provided by using TCG mechanisms, we go on to discuss two distinct approaches to using these mechanisms to enhance the security of P2P networks. The distinction arises from the nature of the "root of trust" for DAA in each version.

In the first version, the root of trust for DAA is assumed to be provided by a credential issued in a controlled manner at the time of manufacture of the platform by one of a small number of platform manufacturers. This model is matched to the widespread deployment of TCG technology on end-user platforms and is most interesting when it is exploited to secure traditional P2P networks. In this context, our research has a quite unusual implication. TCG is often portrayed as a technology which has the potential to provide mechanisms to control the spread of digital content. But in our architecture, it becomes a mechanism that can be used to enhance the security of, for example, a P2P file-sharing network distributing content, some of which may be copyrighted. The security provided does not prevent a copyright owner from purchasing a TCG-compliant platform and joining the network, but it does prevent casual eavesdropping on network communications, and gives a degree of anonymity (strictly, pseudonymity) for the users. While it is still possible for copyright owners or their agents to flood the network with poor quality or bogus material through pseudospoofing, the cost of doing so quickly becomes prohibitive as this would require the use of multiple TCG-compliant platforms in order to claim multiple identities. As we shall report in Section 2, such attacks have been well documented as a mechanism by which the value of P2P file-sharing networks can be reduced. In addition, the originators of such material can be pseudonymously identified and barred from the network, using, for example, a reputation mechanism founded on the stable pseudonyms assured by the use of TCG mechanisms. Going further, our DAA-based access control mechanism can be used to add strict controls on who can access the network. In the limit, completely private P2P networks can be constructed, with peers restricting access to a closed group of pseudonyms.

At this point it becomes clear that there is something of a moral dimension to our work. Since the advent of Napster, the distribution of copyrighted materials via P2P networks has soared. Organisations like the RIAA have attempted to prosecute individuals alleged to be infringing their members' copyrights. If implemented, the work presented here might make this endeavour more difficult. It might even force copyright holders to engage more quickly with new business models for content distribution, an outcome that we see as being inevitable in the longer term. We do not wish to become embroiled in the debate over the rights and wrongs of copyright enforcement and copyright abuse in P2P networks. Nor do we wish to provide tools allowing paedophiles and other similarly unsavoury groups even easier access to private networking capabilities than they already have. Yet it must be recognised that the technology exists and may soon be widespread. It seems better to examine the full range of ways in which it might be used (and abused) than just to ignore these possibilities.

The second version of our security architecture is more commercially-oriented. Here, the

root of trust for DAA is a company offering content or information services to customers who are willing to pay for the service. Now DAA can be used to register customers, authenticate registered customers, and prevent customers sharing registrations. The accesses made by a particular customer (or rather platform) to different service providers are unlinkable because of the properties of DAA. This kind of usage of DAA has already been anticipated in a presentation given by Camenisch [1]. Our work can thus be seen as filling out the details of this approach by specifying which TCG commands need to be issued in order to register and authenticate customers. But, in addition, we present a P2P twist on this fairly routine application of TCG. We propose that registered customers can act as nodes in a P2P network for distributing the content owned by the service provider. Because of the trust properties of a TCG-compliant platform, the service provider can allow this to happen without fear of losing control of its content. At the same time, the service provider can reduce its requirements for storage and bandwidth, instead relying on peers to do this work. One potential incentive for peers to become content distributors could be a reduction in access fees; further possible incentives are discussed in Section 6.2. We also discuss mechanisms by which the service provider can continue to collect revenue from customers who obtain their content from network nodes rather than the service provider itself.

We close this chapter with a discussion of some of the issues and open problems raised by our work. We also point to areas which are opened up by our work and which seem ripe for further exploration.

2 Overview of Security Issues for P2P Networks

2.1 Introduction

Before discussing the security issues, we provide a short overview of P2P networks.

There is a multitude of definitions for the concept of P2P networks in the literature. The one we prefer here is taken from [23]:

> *A peer-to-peer network is a class of systems and applications that employ distributed resources to perform a critical function (usually in a decentralised manner).*

Resources here could be computational power, data, or network bandwidth, while critical functions could include data or information sharing, distributed computing, communications or collaborative working. Within this broad definition of P2P networks, one can include systems such as Napster and Gnutella as well as systems less traditionally thought of as being P2P, such as eBay. While eBay itself is very much a centralised architecture, the reputation system within eBay exhibits P2P aspects, with peers contributing to the reputation of other peers.

Despite the initial and still main use for content sharing, P2P networks have come a long way in a short time and are gradually being adopted in commercial and corporate applications such as network storage [21], content distribution, web caching [19] and application level multicast [3]. P2P networks represent something of a paradigm shift from the traditional client-server model. The P2P approach can bring many benefits: scalability, efficiency, fault resilience and, of course, reduced reliance on central servers. It is through the entities in these networks, called *peers*, that resources are replicated and shared.

2.2 Security Issues

Whilst many aspects of P2P networks have been thoroughly researched, security within these networks still remains a challenge. It is not our intention to exhaustively review security issues for P2P networks here. Indeed, good overviews of this topic can be found in [8] and [35]. Rather, we focus on identifying some key security issues in P2P networks and pointing towards some of the approaches to solving them that have been identified in the literature.

The security architecture [18] associated with the ISO/ITU Open Systems Interconnection (OSI) reference model serves as a useful framework for assessing security issues in networks. According to [18], a secure system is governed by the set of security services it provides, and the mechanisms put in place to cater for these services. The set of potential security services in [18] are divided into five main classes:

- Confidentiality,
- Integrity,
- Authentication – including data origin authentication and entity authentication,
- Access Control, and
- Non-repudiation.

Additional services not explicitly included in [18] could be added to this list. Accountability is an important service in networks where users are to be charged for their use of resources. Anonymity in various forms is often cited as being desirable, especially in the context of P2P networks. Anonymity can refer to obscuring all identifying information of an individual, or it can refer to making the actions of a particular individual unlinkable. Availability is sometimes considered separately from security. Since many malicious attacks on networks are directed at reducing availability for legitimate users, we explicitly include it here as a security service. However, we do not consider it in any further detail – a good summary of the issues can be found in [8]. Any particular network may require a combination of all, some or even none of these services.

In traditional client-server systems, users are typically identified with a user account, and platform or system-specific controls can be applied to these accounts to provide security mechanisms. Security services such as access control and accountability can be implemented in this manner, with the accounts providing a form of stable identity. Other services such as authentication and non-repudiation clearly also rely on the establishment and preservation of stable identities. Moreover, if we want to establish cryptographic keys for confidentiality and integrity of data in transit, then entity authentication of one or more of the communicating endpoints is usually necessary for security. Often, entity authentication is enabled using Trusted Third Parties (TTPs). For example, one might rely on a Public Key Infrastructure (PKI) and the certificates issued by a Certificate Authority, or one might use the services of a dedicated TTP as in Kerberos. Thus most (though not all) the generic security services we might wish to provide are reliant on the provision of stable identities.

In a P2P environment, there are by definition no centrally-administered accounts or trusted third parties who can provide assurances as to the identities of entities in the network. Instead, each peer is typically identified based on a *handle* or *pseudonym* that is selected by the peer for itself. In most P2P networks, there exists no standard registration procedures for these pseudonyms, and indeed entities can claim multiple pseudonyms, including those of other entities in the network. The use of pseudonyms provides a degree of anonymity for peers

and eases discovery and routing of resource queries in P2P networks, but is clearly in conflict with the stable identities that are needed, as noted above, to provide many other security services. This problem of providing entity authentication in the face of untrusted networks of peers seems to us to be a central challenge in providing wider security services for P2P networks. Having identified this challenge, we now go on to make a more detailed examination of the security issues in P2P networks, and their relationship to the identity/authentication issue.

2.2.1 Authentication and Pseudospoofing

Pseudospoofing, a term coined by Detweiler [11], refers to a peer creating and handling more than one pseudonym at once. Potentially, an attacker might manipulate tens or even hundreds of pseudonyms to his advantage. An attacker might also make use of the pseudonym of an existing peer, preferably one with a good reputation. This type of attack has been named *ID stealth* in the literature [7], but we prefer the term *pseudotheft*. An attacker who engages in pseudospoofing or pseudotheft in a P2P network may do so in order to manipulate a reputation mechanism in place in that network. A peer, having gained a bad reputation, can discard the corresponding pseudonym and re-join the network with a fresh pseudonym. In on-line auction systems, a peer can use a pseudonym to generate false bids, an attack known as *shilling*. In general, a peer can create a number of pseudonyms, build up the reputation of one or all of the pseudonyms using the others, and then make use of the now reputable pseudonyms to persuade other peers to trust him. Clearly any P2P network attempting to rely on a reputation system for building trust in peers would need to address the problems of pseudospoofing and pseudotheft. It is evident that pseudotheft can also lead to loss of confidentiality. It has been argued [14] that in any P2P system without a centralised point of trust, such attacks on identity are endemic and can never be effectively combatted.

Lest these attacks seem theoretical, let us note that there are many concrete cases citing the issue of pseudospoofing as a potential problem. We briefly discuss two examples. The reputation system used in eBay, essentially a P2P system, has constantly been under pseudospoofing attack. A particularly well-documented case is that of the philatelic frauds perpetrated by the Saratoga gang on eBay.[1] The Recording Industry Association of America (RIAA) and their agents have been accused of setting up bogus identities in order to spread poor quality music files in certain P2P file sharing networks, notably Kazaa and networks based on Gnutella, with the alleged aim of disrupting these networks to prevent sharing of copyrighted materials.[2] These activities amount to attacks on the authenticity of resources on the networks.

The key aspect to note in the above examples is the relative ease with which peers can create, use and abuse identities. In the absence of strong registration and entity authentication procedures, it becomes hard, if not impossible, to enforce stable identities or pseudonyms. While the problem is widely recognised, research directed towards preventing pseudospoofing has been sparse, perhaps reflecting the truly decentralised nature of P2P networks. For example, the problem is discussed in [7], but it is simply assumed there that identifiers are

[1] An interesting account of this case, "The Saratoga Fakes", has been compiled by the Stamp Collectors Against Dodgy Sellers (SCADS) institute, `http://www.scads.org/alterations/Saratoga.htm`.

[2] See, for example, the website `http://www.riaamix.com`, which jokingly hails the "music" in these corrupted files as a new art form. Patents owned by Overpeer (`http://www.overpeer.com`) specifically refer to this kind of disruptive activity.

tamper-resistant. A statistical, quorum-based approach to building a PKI suitable for supporting P2P security services, including entity authentication, was proposed in [10]. Another important contribution to a potential solution to this problem is given by Advogato[3] where trust relationships are modelled as a directed flow graph. Trust between peers is modelled as the presence/absence of an edge in the graph and the amount of weight assigned to this edge determines how much one peer trusts another. How trustworthy a peer is in relation to the graph as a whole is calculated as the maximum flow that can be pushed from a source (trusted) peer to another peer. When a peer joins the graph it is isolated from other peers and so there is little reason to trust it.

2.2.2 Accountability and Reputation Mechanisms

In the P2P context, accountability means the ability to hold peers responsible for their actions and the resources that they share. This aids in achieving efficient utilisation of resources, discourages peers from consuming resources whilst only providing nominal resources in return (a practice widely known as free-riding), and gives peers protection against fraud.

Despite these known benefits, it is hard to ensure accountability due to the inherently autonomous and transient nature of P2P networks. A peer can join and leave the network at any time. Network topologies are dynamic and it is uncommon to have a known history of all peers, nor are there any contractual obligations between peers. Most obviously, anonymity and accountability requirements are in clear opposition. Moreover, these problems are exacerbated by the lack of stable identities in P2P networks.

There are various schemes in the literature aimed at addressing the issue of accountability in P2P networks. A common approach is to use some kind of reputation mechanism to build trust in P2P entities and the resources they hold. The rationale for this approach is that knowledge that peers will consider each other's past interactions while pursuing present and future transactions constrains peer behaviour in the present, thus imposing accountability in the form of non-abuse of resources [28].

A classic example of such a reputation mechanism is the web-of-trust approach adopted in PGP [36]. Here, a user can calculate a trust level associated with another user's public key, where the calculation depends on a number of factors, including which other users are prepared to trust that key, and how well they themselves are known to the relying party. The scalability and applicability of this kind of approach for large and widely distributed P2P networks has been questioned in [10], where an alternative approach based on distributed PKI was suggested. Another interesting approach, based on micropayments, was proposed in [13]: users are made accountable for their use of resources via micropayments. The micropayment tokens are generated through work performed by the users. The P2P network Mojo Nation[4] uses its own currency called 'mojo' to handle micropayments between peers. On the other hand, Free Haven [24] requires participants to provide storage space for other peers to use, a form of payment in kind for accessing the network.

However, we have already seen that these reputation mechanisms are open to manipulation through pseudospoofing and other attacks.

[3] `http://www.advogato.org`
[4] `http://sourceforge.net/projects/mojonation`

2.2.3 Anonymity

Anonymity in a P2P context, in its strongest sense, means that there should be no mechanisms that link peer identities, aliases, actions or responses within a network. Anonymity greatly appeals to the P2P user base because it not only offers them privacy protection, enabling facilities such as censorship resistance and freedom of speech, but also the ability to break copyright laws or publish libellous material without fear of retribution.

Anonymity for P2P networks has been classified into four distinct categories [24] – *author anonymity* (which user created which resource), *server anonymity* (which peers store a particular resource), *reader anonymity* (which peers access which resources) and *document anonymity* (which resources or documents are stored at a given peer node). As we have already noted, the provision of these forms of anonymity conflicts with other security goals and hampers the design of strong reputation mechanisms. Moreover, providing complete anonymity usually impacts on network performance and routing [6, 22]. As examples, the P2P networks Freenet[5] and Publius[6] have made efforts to provide anonymity. Freenet, in particular, encrypts data between its nodes and routes messages through other nodes. In the process, it is made difficult to determine which node is requesting data and what the data contents are. Also, peers have to contribute a portion of their disk space to the network, but peers do not know what is stored in their local data stores. These techniques are, in general, adapted from existing anonymity techniques such as MIX networks [4, 27], crowds [26] and onion routing [30].

An absence of anonymity implies a loss of privacy. Therefore present systems typically employ a middle ground whereby each peer within the network is recognised by a pseudonym. Here, the actions associated with a particular pseudonym can be linked, but there is not a strong binding between the pseudonym and an underlying peer identity.

2.2.4 Access Control

Another security problem faced by peers within a P2P network is access control. Within a P2P environment, this would mean restricting access to authorised peers, for example, peers who have paid for the queried resources. This would certainly be desired by companies managing intellectual property and digital rights, but may also be of interest in more traditional P2P networks. The (deliberate) absence of such controls has led to widespread abuse of copyright laws and a consequent fusillade of litigation against companies making P2P-based software and individuals operating on P2P file-sharing networks. However, it is certainly not the case that access control is needed in every P2P network. A careful analysis would be required to evaluate what impact an access control mechanism would have on peer reputations. For example, a peer restricting access to queried resources may get a bad reputation with other peers. Moreover, building robust access control mechanisms once again depends on authentication: without stable identities, traditional access controls cannot be implemented. One piece of relevant research can be found in [15], which proposes the distribution of the access control problem within communities of peers. These communities, in turn, are governed by a set of L1 (level 1) peers who maintain the security architecture. They authorise L2 (level 2) peers and protect access to resources within that community. However, this notion of a hierarchial P2P network runs counter to the decentralised nature of numerous P2P network

[5] `http://freenet.sourceforge.org`
[6] `http://cs1.cs.nyu.edu/~waldman/publius/`

in current use.

2.2.5 Confidentiality and Integrity

Confidentiality and integrity can be classified broadly as applying to storage (data at rest) or communications (data in transit). In the context of P2P networks, the former refers mainly to resources stored at peers, while the latter refers to transactions that occur between peers, for example, resource queries and replies, distribution of reputation information, and so on. Attacks on confidentiality and integrity can be achieved by a variety of means. For example, it may be possible to manipulate the routing protocols used in a P2P network so that data of interest to an attacking peer is routed via that peer.

A common method for ensuring integrity of resources in transit and at rest in P2P networks is to calculate and append MAC values or digital signatures to the resources. While an integrity service guarantees that a resource has not been altered (either in transit or while at rest), it may say little about the original source of that data. Questions of that nature can be settled using a data origin authentication service in combination with, for example, a time stamping service. This is referred to as file authenticity in [8], but we prefer the terminology of [18].

Confidentiality and integrity are not often discussed in the context of P2P security. Some research and implementation has already taken place in certain networks like Groove[7] but these services are optional there. If one regards PGP as providing a secure P2P messaging system, then it provides a good example of the careful implementation of confidentiality and integrity services in a P2P context. Confidentiality and integrity are clearly fundamental to security, and we believe they are set to become more important with the emergence of commercial P2P networks. The requirement to protect valuable resources at rest and in transit is likely to outweigh the processing and management overheads associated with these services, and will be a key enabler for P2P e-commerce.

Once again, a barrier to the development and deployment of these security services for data in transit in P2P networks is the lack of stable identities. The problem is not that one cannot implement the services using well established encryption and MAC mechanisms, but that there is little point in doing so if peers do not know with whom they are establishing communications. Otherwise, a peer may fall victim to man-in-the-middle attacks or send sensitive data to unintended recipients. Once again, authentication of peers is a vital precursor to providing security services in P2P networks.

2.3 Summary of Security Issues

Research on P2P security has covered a variety of problems and is summarised here and elsewhere [8, 35]. We have identified what we believe to be a fundamental issue which must be addressed if better than trivial security is to be added to P2P networks: the lack of stable identities and the inability to provide strong authentication services, which in turn underpin the security of reputation systems, accountability, access control, confidentiality and integrity services in P2P networks. We have also seen that full anonymity in P2P networks clashes with other security services. A reasonable compromise is to work with pseudonyms, shielding the binding between on-line and "real-world" identities, but to find methods to prevent pseudospoofing and pseudotheft attacks.

[7] `http://www.groove.net/home/`

3 Overview of Trusted Computing

3.1 Background

Before we move on to the main focus of this chapter, it is necessary to reintroduce certain core aspects of TCG technology. In particular we review the mechanisms involved in integrity measurement and storage as well as that of Direct Anonymous Attestation (see also Chapters 3 and 5). In the context of the TCG specification, integrity measurement and storage fall under the remit of the Root of Trust for Measurement (RTM) and the Root of Trust for Storage (RTS) respectively. These two roots of trust form the basis for the trustworthiness of a platform and are responsible for the acquisition and storage of integrity metrics within a platform. The rationale behind measuring and storing integrity altering events is that it allows a platform to transition into any number of possible states which can be made visible to interested external parties, but the platform is unable to falsely adduce its current state.

3.2 TCG Integrity Mechanisms

Examining the TCG integrity measurement and storage mechanisms at a high level, we note two structures of particular relevance to this chapter, namely Platform Configuration Registers (PCRs) and the Stored Measurement Log (SML), alternatively referred to as the Event Log. In the context of a TCG enabled platform, it is the SML that is responsible for maintaining an ordered database of integrity changing events. Each PCR has responsibility for the maintenance of a cumulative digest of one or more of these events. The data held in a PCR, in conjunction with the relevant portion of the SML, is used as evidence to attest to a current platform state.

Each PCR is an eight byte storage area held within a Trusted Platform Module (TPM). A PCR contains a representative digest of a number of measured values, typically consisting of embedded data or program code. The basic mechanisms of PCR usage have already been covered elsewhere in this book so we shall not dwell on them here. However, for clarity we will briefly review one aspect, as it is integral to our discussion – that of PCR updates.

PCR updates occur by appending the new event being measured to the existing value contained in the register, and then taking a SHA-1 hash of this value. This can be expressed programatically as follows:

$$\text{PCR}[n] \leftarrow \text{SHA-1}(\text{PCR}[n] \parallel \text{measured data}).$$

Here n is the number of the PCR being updated. This operation is commonly referred to as extending the digest. The use of this approach guarantees that related measured values will not be discarded as the previous register value is incorporated into the new register value. It also ensures that updates to a PCR are not commutative: the order in which events occur and the measured values they contain both affect the value of the stored digest. As it is necessary to preserve the complete event histories for a particular PCR, and the specification mandates that there be at least sixteen PCRs within a TPM, there is the potential for the SML to grow quite large. For this reason the SML may be stored outside the TPM in non-shielded memory.

The PCRs and the SML are intimately related. Without the representative digests contained in PCRs, the events maintained by the SML are meaningless (as there is nothing to stop someone from generating false events in the log). Without the relevant portion of the

SML, a PCR digest is devoid of any meaning (as the context from which the digest was generated would be lost). Consistency between the two structures is preserved by the fact that PCRs are maintained internally to the TPM and that PCR extension operations are only permitted via TPM protected capabilities. These PCRs provide evidence of attempted tampering of the log, since the sequence of events tied to a specific PCR can be extracted from the SML, rehashed and compared to the actual value contained in the PCR. Thus, the ability to securely update PCR registers is integral to the trustworthiness of a platform.

Now that we have a better understanding of the basics of how integrity measurements are taken and stored within a trusted platform, we can move on to examine the topics of integrity reporting and attestation.

3.3 Platform Attestation

When a verifier wishes to inspect a host platform's configuration, it may request a specific set of PCR values which, as discussed above, contain a representative digest of a sequence of events within a platform. Assurance as to the validity of a PCR value is provided by means of a signature on the register value produced by the TPM. The TPM uses the private component of a key pair called an Attestation Identity Key (AIK) to perform the signing operation. Within a TCG conformant platform, AIK key pairs are used as aliases for another key pair called the Endorsement Key (EK). There is no prescribed limit on the number of AIKs that can be used within a platform. This provides an anonymity mechanism, whereby the TPM can use different AIKs each time it signs a PCR value, making those attestations unlinkable. The reason for introducing this anonymity mechanism is that the EK uniquely identifies the platform, and possibly the owner of the platform.

The actual TCG attestation protocol is illustrated in Figure 1 and outlined below.

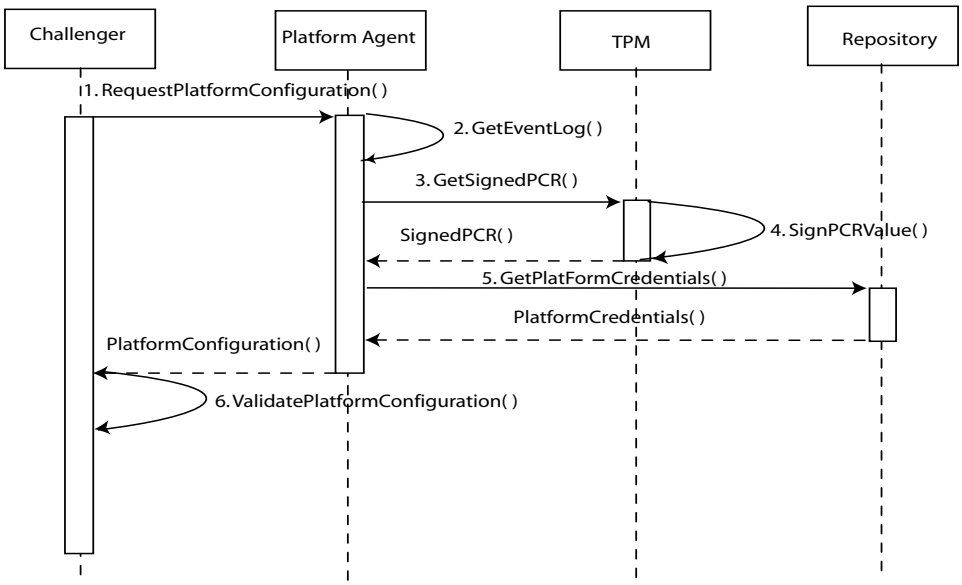

Figure 1: TCG Attestation Protocol [34, p.9]

1. A Challenger wishes to inspect one or more PCR values on a platform.
2. A Platform Agent gathers the relevant SML entries corresponding to the PCR values requested.
3. The TPM sends the Platform Agent the requested PCR values.
4. The TPM attests to the values contained in the requested PCRs by producing a signature using an AIK private key.
5. The Platform Agent assembles credentials that vouch for the TPM. The signed PCR values with their corresponding SML entries and relevant credentials are returned to the Challenger.
6. The Challenger verifies the supplied data. The measurement digest is computed from the returned SML entries and compared with the signed PCR values. The platform credentials are evaluated and signatures checked.

As shown in the Figure 1, a Challenger obtains a portion of the SML (possibly all of it) and one or more PCR values, allowing it to confirm that the platform is indeed in a specific state by examining event sequences. The way in which a platform proves that it is in possession of the relevant credentials differs in versions 1.1 and 1.2 of the TCG specification. In version 1.1, it was envisaged that the platform would obtain a credential from a trusted third party known as a Privacy CA. The credential would come in the form of digital certificate issued by the Privacy CA on a specific AIK, with the Privacy CA being aware of the binding between AIK and EK. Thus the privacy properties of the scheme would depend on the trustworthiness of the Privacy CA, and, to an extent, the commercial success of Trusted Computing would depend on the emergence of suitable organisations offering Privacy CA services. Version 1.2 still supports this Privacy CA mechanism, but also introduces a new approach called Direct Anonymous Attestation (DAA). DAA, the subject of the following section, uses more sophisticated cryptographic techniques to ensure the privacy of users, without introducing the requirement for special trusted third parties.

3.4 Direct Anonymous Attestation

We provide here a brief overview of the key features of DAA that we will use in building security for P2P networks. The full cryptographic details can be found in [5], while the TCG specification [34] contains implementation specifics. We also refer the reader to Chapters 3 and 5.

Before a platform can provide attestations to a Challenger (called a *verifier* in [5]) concerning its current configuration state, it is necessary for the platform to go through a *join phase* where it obtains a credential from a trusted third party called an *issuer*. In practice, the part of the issuer might be played by the platform manufacturer or by a third party who wishes to offer services exploiting TCG features.

During the join phase the platform asks to become a member of the group of platforms to which credentials have been issued by that issuer. During this process, the platform proves that it has knowledge of a specific TPM-controlled, non-migratable secret value f and is authenticated to the issuer via its EK. If the issuer accepts in this step, then it provides the platform with a credential in the form of a *Camenisch-Lysyanskaya (CL) signature* [2] on the

secret value f. It is this CL signature that will later enable the platform to convince a verifier that it has previously successfully completed the join phase with a particular issuer. Note that the issuer does not actually learn f in this process. Nor does the TPM actually store the value f – due to the limited resources available within a TPM, f is generated by the TPM every time it is needed from a seed value in conjunction with some information supplied by the issuer.

In order for a platform to obtain a credential for its secret value f, the platform must reveal a pseudonym N_I of the form ζ_I^f, where ζ_I is derived from the issuer's name and is an element of some suitable group. By maintaining a history of pseudonyms that it has seen, the issuer can check if credentials have already been issued to this TPM. The issuer can also check for rogue platforms at this stage, by testing if $N_I = \zeta_I^f$ corresponds to an f appearing on a blacklist.

Once the join phase is complete, the TPM (with the help of the platform) can use the DAA-Signing algorithm to prove to any verifier that it has an appropriate issuer credential (in the form of a CL signature), and at the same time, sign a message m using the secret value f. Here, m is typically the public component of an AIK key pair, but, according to [5, Section 4.6], it can also be an arbitrary external 160-bit string. In the former case, the output of the DAA-Signing algorithm amounts to an attestation that the TPM is in possession of a valid issuer credential and has a particular AIK. This gives the verifier a similar assurance concerning that AIK's linkage to a particular platform to that gained using the Privacy CA approach described in the previous section. In turn, this allows any PCR value signed by the particular AIK to be checked as being genuine by the verifier. Corresponding to the DAA-Signing algorithm is a DAA-Verify procedure that is executed by the verifier.

In the DAA-Signing algorithm, the platform is only identified by a pseudonym. As in the join phase, this pseudonym is of the form $N_V = \zeta^f$, where now the selection of ζ determines the anonymity properties of the attestation process. The value of the base ζ will typically be chosen by the verifier. In this case, if the verifier fixes on a choice of ζ (deriving ζ from its name, for example), then this will also fix the value of the pseudonym N_V used by a particular platform. This will allow all the transactions involving a particular TPM to be linked. It is worth pointing out here that the secret value f is non-migratable, so an f value should not be able to leave the confines of a TPM, and hence only the TPM that generated that value should be capable of using it. If a value of f should escape the TPM, then the blacklisting approach used during the join phase can be used, provided verifiers have access to the list of rogue f values. The value of ζ might also be selected at random by the platform for each transaction with the verifier – this would allow the platform's transactions to become completely unlinkable.

4 P2P Pseudonymous Authentication Using Trusted Computing

In this section, we consider how the use of stable pseudonyms can be enforced in any P2P network in which each peer is TCG-enabled and has appropriate DAA credentials. In subsequent sections, we will examine how this basic property can be leveraged to provide security services in P2P networks.

The mechanism for enforcing stable pseudonyms is simple. We assume only that the P2P network has a particular name, that is, a unique identifier. The pseudonyms used in our

network are strings of the form $N_P = \zeta^{f_P}$, where ζ is derived from the network name by applying a suitable hash function to produce a group element, and f_P is the secret associated with the TPM located on the peer P claiming the pseudonym.

Whenever a peer claims a particular pseudonym, the peer verifying that claim can challenge the claimant to supply a DAA signature on, say, a random message M. Later, we will see how to enable other security services by selecting M to include additional values. The verifying peer can check, using the DAA-Verify algorithm, that the claimant has a valid credential supplied by a particular issuer and can be convinced that the value f_P used in forming the claimed pseudonym N_P is the same as the one claimed at the time of credential issuance. The use of a random message M prevents replay attacks in which a rogue peer captures and replays credentials. Through these checks, the pseudonym N_P that can be claimed by peer P is fixed, and is determined as a function of the network name and the particular f_P certified during the joining phase. In summary, we have built an *pseudonymous authentication mechanism* from the DAA algorithms. Through this mechanism, peers can be authenticated by other peers, but platform identities are not revealed in the process.

To make this a workable method for enforcing stable pseudonyms, we need to ensure that a particular platform will only ever obtain one credential from one of a set of authorised issuers that are recognised by the verifying peer. Otherwise, it may be difficult to prevent a peer obtaining multiple credentials for different f values, from one or more issuers, and using these to produce multiple pseudonyms. One mechanism for ensuring that a platform can only ever obtain a single credential is to assume that an initial credential is issued to the platform in a controlled fashion at the time of manufacture. The set of authorised issuers is then the set of platform manufacturers, and the peer software can be configured to insist on seeing a credential issued by one of these manufacturers. Here, then, the manufacturers become the root of trust for DAA. This is a likely scenario in the deployment of trusted computing. We explore this situation further in Section 6.1. We relax the assumption about issuance of single credentials in our second model for the use of DAA in P2P networks, described in Section 6.2.

We now examine the extent to which our approach prevents pseudospoofing and pseudotheft attacks. We note that, if our condition on credential issuance is met, then a peer cannot claim multiple pseudonyms. Nor can a peer claim the pseudonym of another peer. Thus these attacks, so destructive in P2P networks, are prevented. The usual blacklisting mechanisms can be used by peers to detect the use of a rogue TPM, provided the relevant information can be distributed and kept up-to-date. Of course, nothing in our approach prevents an attacker from purchasing multiple TCG-compliant platforms and using these to create multiple pseudonyms. However, this kind of attack has an economic cost for the attacker.

It is clear that, by enforcing the use of pesudonyms, a given peer's actions on the network become linkable, so our approach does not offer a full level of anonymity. We have already established that this is necessary if other security services are required. Moreover, the use of DAA never reveals the platform identity (in the form of the endorsement key) to verifying parties, so the peer's actions cannot be linked to a particular TPM. Notice too that the pseudonym used during the join phase is different from that revealed during DAA-Signing (because $\zeta_I \neq \zeta$ in general). Thus the pseudonyms do shield the actions of a peer effectively. We also note that the pseudonyms we propose for use do not relate to particular individuals; rather they are linked to particular platforms (more precisely, TPMs).

A given peer may wish to operate in more than one P2P network. Because of the use of

the network name in determining the value of ζ used to form pseudonyms, we obtain the nice feature that a peer's actions in different networks will remain unlinkable.

Finally, we note that a somewhat less attractive authentication mechanism can be built using the legacy Privacy CA approach. The idea is to extend PCRs using nonces exchanged between peers as measured data. The PCRs can be signed using AIKs, which are in turn signed by the Privacy CA. The signed nonces can be exchanged by interleaving two runs of the TCG attestation protocol. We omit the details since they are closely related to the techniques introduced in the next section.

5 Securing P2P Networks Using Trusted Computing

In this section we will demonstrate how a range of security services can be built from our DAA-enabled authentication mechanism and the stable pseudonyms which it enforces.

5.1 Access Control and Accountability

As outlined in Section 2.2.2, access control is particularly problematic in large unrestricted P2P networks. Given the above authentication mechanism built from TCG mechanisms, it would be trivial to implement a simple access control mechanism based on access control lists. In this case, an access decision could be made based on the peer pseudonym presented to (and verified by) the peer granting access to resources. Clearly a more sophisticated and application-dependent access control system could be implemented on top of the stable pseudonyms.

As we have reported in Section 2.2.2, accountability and robust reputation models are hard to build in the presence of pseudospoofing and pseudotheft. Now that we have the ability to enforce stable pseudonyms in a P2P network, it becomes possible to build robust reputation systems that will not be undermined by pseudospoofing and pseudotheft attacks. Such a reputation system provides a means to hold each peer accountable for its actions. We note that the deployment of reputation systems seems to be a necessary precursor for the development of P2P e-commerce as envisioned in [10]. The description of specific reputation systems is beyond the scope of this chapter.

5.2 Authenticated Key Establishment and Secure Channels

Whilst authentication (even at the level of pseudonyms) is a very useful service to provide in a P2P network, its usefulness is limited if it cannot be extended to provide protection for an entire communication session between peers. We address this next, showing how our authentication mechanism can be extended to an authenticated key establishment protocol. From there, it is a short step to integrating our DAA-based mechanisms with SSL and IPSec, these being two widely used existing methods for providing secure channels for communicating entities.

5.2.1 Two Generic Approaches

Suppose we have two peers P and Q, with pseudonyms N_P and N_Q, in a particular P2P network. We assume these peers wish to authenticate one another[8] and establish keying

[8]The authentication need not be mutual, and the alterations to our methods needed in this case are trivial.

material for protecting further communications.

In the simplest form of our approach, the peers exchange random nonces R_P, R_Q, ephemeral Diffie-Hellman values g^{x_P}, g^{x_Q} (where g is a generator of a suitable group negotiated in advance or by peer software configuration), and signatures on the 160-bit hash[9] of the message

$$M = N_P \| N_Q \| R_P \| R_Q \| g^{x_P} \| g^{x_Q}.$$

The signatures are obtained by each peer using their TPM and platform to execute the DAA-Signing algorithm on the hash of M. Using the DAA-Verify algorithm, each peer can check that the other has a valid credential supplied by a particular issuer and can be convinced that the values f_P, f_Q used in forming the claimed pseudonyms N_P, N_Q are the same as those claimed at the time of credential issuance. Here, we are essentially re-using the pseudonymous authentication mechanism from Section 4, but with M made up of a longer string including pseudonyms, nonces and Diffie-Hellman values. If both signatures are confirmed by DAA-Verify, then the parties can safely compute common secret keying material $K = g^{x_P x_Q}$, and then derive keys for MAC and symmetric encryption algorithms from K.

Here, we have used the facility that a platform can ask the TPM for the DAA signature on any message M. However the TPM may be configured to only execute DAA-Signing on AIKs. If this is the case, we can use an alternative (but slightly less efficient) approach involving PCRs and AIKs. Each peer can use the string $M = N_P \| N_Q \| R_P \| R_Q \| g^{x_P} \| g^{x_Q}$ as measured data to extend a PCR, using the process outlined in Section 3.2. Then each peer can request its TPM to produce an attestation to the PCR value, which takes the form of a signature on the PCR value produced using the private component of the peer's AIK. Next, at each peer, the DAA-Signing algorithm can be used to sign the peer's AIK. Finally, the peers can exchange the DAA signatures on their AIKs and the AIK signatures on the string M. Checking these signatures provides the necessary authentication for the Diffie-Hellman key exchange. This approach can be realised in practice by first extending the PCRs and then interleaving two runs of the TCG attestation protocol outlined in Section 3.3.

We now go on to investigate a third approach, better suited to integration with existing secure protocol suites, in Section 5.2.2. After that, we examine which specific TCG commands are needed to implement our ideas.

5.2.2 Integration with SSL/TLS and IPSec

We have outlined two mechanisms by which DAA can be bootstrapped to build secure communications between pseudonymously authenticated endpoints. Our motivation here is to build on existing protocols to achieve the same aim, still using DAA and attestation of PCRs as a basis for authentication, but re-using protocol components where possible. We focus on SSL/TLS and IPSec as starting points.

SSL/TLS [12] is a protocol suite running on top of TCP and providing a reliable, end-to-end secure communications service. Traditionally entities engage in the SSL/TLS Handshake Protocol to establish keying material and authenticate. This keying material is then used in the SSL/TLS Record Layer Protocol for data integrity protection and encryption. SSL/TLS supports a wide variety of key exchange methods, including both anonymous and ephemeral Diffie-Hellman exchanges, supported by digital signatures, as well as key establishment based

[9] The TCG specification limits externally generated messages that are to be DAA signed to be at most 160 bits in length.

on RSA public key encryption. Both unilateral (server to client) and mutual authentication are allowed.

Either one or both parties in SSL/TLS will have a key pair that is used in the Handshake Protocol. Validity of these public keys is ensured through the verification of X.509 certificate chains back to a trusted root. We show how to replace this process with pseudonymous authentication in a TCG-enabled P2P architecture. The main idea, once again, is to extend a PCR, this time using the public key(s) as measured data. By signing the extended PCR using an AIK, and then signing the AIK using the DAA-Signing algorithm, a chain of signatures extending from the SSL/TLS public key(s) to the DAA issuer's public key can be constructed. Thus, in this approach, the SSL/TLS root CA would be represented by the issuer of DAA credentials.

One complexity here is that at least one of the signatures in this chain is not of the type traditionally seen in SSL/TLS, namely the Camenisch-Lysyanskaya signature representing the issuer credential. Another complexity is that the chain of signatures are not arranged in a neat X.509 certificate chain. For this reason, it may be simpler to just use self-signed X.509 certificates in the SSL/TLS Handshake Protocol and follow this protocol run with a second phase comprising an exchange of DAA credentials and PCR attestations using two runs of the TCG attestation protocol. Ciphersuite negotiation in SSL/TLS would determine which keys needed to be incorporated into PCRs in the TCG attestation protocol. Nonces exchanged in the SSL/TLS Handshake Protocol should be included as part of the measured data in the PCR extension step to bind the SSL/TLS key exchange and pseudonymous authentication steps.

This approach is sufficiently flexible to support any of the SSL/TLS standard key exchange methods. As options, the issuer names in the self-signed certificates could be the peer pseudonyms N_P, N_Q and the subject name field could be a meaningful handle that the peer (user) wished to be known as on the network. This last option would help to address the issue of associating a meaningful name with a platform's pseudonym, an issue we will return to in Section 7.1.2. All of this could be made transparent to the users in the same way that SSL/TLS hides the intricacies of its protocol in web browsers today.

Using similar ideas, we can also build on the IKE protocols within IPSec, as specified in [17], to establish secure channels between peers. Further traffic between peers can then be protected using IPSec's AH and ESP protocols in appropriate combinations. In the simplest version, we can use either of the generic key exchange mechanisms from Section 5.2.1 to establish a shared key, then use the shared key version of IKE Phase 1 to establish a secure ISAKMP channel over which further Security Associations can be exchanged. Alternatively, any of the IKE Phase 1 versions using public key techniques can also be supported. We can use the technique of extending PCRs, with measured data being replaced by the relevant public keys, followed by TCG attestations, to establish authenticated public keys at the communicating endpoints. This can then be followed by the appropriate version of IKE Phase 1, again using self-signed certificates to simplify certificate processing, and using DAA pseudonyms N_P and N_Q as initiator and responder identities. After this, IKE Phase 2 can be used to establish multiple further SAs containing keys for use in AH and ESP protocols between the peers.

Some care needs to be taken with these approaches to ensure that the IKE and TCG attestation protocol runs are bound in an appropriate cryptographic manner. We also have not sought to optimise the protocols to any great degree. Much scope remains for developing these ideas further.

5.2.3 Implementation

In order to make our proposals for TCG-based authentication, access control and secure channel establishment mechanisms as concrete as possible, we give here an indication of the specific TCG commands that are needed to implement our ideas.

In order to understand the mechanisms for extending a PCR register we need to examine the event logging mechanism for trusted computing in greater detail. We will base our discussions on the assumption that our P2P application is running on a TCG-conformant platform. The general principles and commands discussed here are transferable to any TCG-enabled device. As we mentioned previously, the SML is not really a log. It is better to envisage it as an array of events in which each entry is in the form of a `TSS_PCR_EVENT` structure [31, p.3]. In a TCG-enabled device it is a `TSS_PCR_EVENT` that provides information regarding individual PCR extension events. The `rgbEvent` parameter in a `TSS_PCR_EVENT` is the one we are interested in, as it is a pointer to event information data, in our case a string formed from cryptographic parameters. It is the TCS (TCG Core Services) Event Log services that are responsible for maintaining the SML. These services are responsible for allowing PCR extension events to be logged and allowing challengers interested in these events access to them.

To incorporate an event into a PCR two things must happen. A call to the `TPM_Extend` command [33, p.114] is needed. This is the command that is responsible for extending the PCR. Then an additional call to a function such as `Tcsi_LogPcrEvent` is needed. This function is responsible for adding the new event to the end of the array associated with a named PCR [31, p.230]. When a challenger wishes to verify a system by examining one or more PCRs (as indicated in step 1 of the TCG attestation protocol) the platform must perform a number of operations to satisfy this request. A call to `Tcsi_GetPcrEventsByPcr` returns an event log of all events in the SML that are bound to a single PCR [31, p.234]; this can be called multiple times depending on the number of requested PCR values. The event log is returned as an ordered sequence of `TSS_PCR_EVENT` structures. Attestation of the chosen PCR values occurs after a call to `TPM_Quote` [33, p.116]. This operation is responsible for providing cryptographic reporting of PCR values, using an AIK key to sign the current value of a chosen PCR.

The `TPM_DAA_JOIN` command [32, p.129] is responsible for obtaining the issuer's CL signature on the DAA secret f, while the `TPM_DAA_SIGN` command [32, p.131] is responsible for running the DAA-Signing algorithm and, through this, proving that the TPM in question does indeed possess valid credentials. By performing the `TPM_DAA_SIGN` command upon the AIK key used in the `TPM_Quote` command, a TPM can prove to the verifier that it is in possession of the private component of the AIK key that was used to sign a requested PCR value.

6 Two Approaches to Securing P2P Networks Using Trusted Computing

In this section, we investigate further the application of TCG functionality (particularly DAA) to securing P2P networks. We consider two distinct approaches here. In the first, we consider augmenting the security of today's "anarchic" P2P networks by exploiting the presence of manufacturer-issued DAA credentials. The second is more tuned to existing commercial

services for distributing content, but we sketch how secure P2P features can be added to these networks, giving potentially significant benefits to service providers.

6.1 Securing Anarchic P2P Networks Using Trusted Computing

As we have seen in the previous sections, DAA can form the basis for a number of security services in P2P networks in which peers are equipped with TCG-compliant platforms. Foremost amongst these services is pseudonymous authentication, from which other useful services (access control, reputation mechanisms, secure communications) can be built.

To achieve this, we have made the single, but vital, assumption that, in any given P2P network, the TPMs embedded in peers are guaranteed to only possess a DAA credential from one of a set of isuers that will be recognised and accepted by other peers. We stress that there can be many issuers of DAA credentials for the platforms in our P2P network, but we need to be sure that each platform only has one credential from one of these issuers, and not multiple credentials from a single issuer or single credentials from each of multiple issuers. Otherwise, misbehaving peers will be able to create and prove possession of multiple pseudonyms.

As discussed above, one way of ensuring this assumption holds is to assume that manufacturers will provide DAA issuer functionality for the platforms that they produce. This seems to be an attractive supposition, since, in reality, customers would gain a privacy advantage if they were to obtain their credentials from their original equipment manufacturer. Either the DAA credential could be embedded into the platform at the time it is shipped to the customer, or it could be obtained at a later stage through a run of the join protocol. A manufacturer can keep track of which EKs it has injected into the TPMs that it has produced and which EKs it has already issued credentials to. Since during the join protocol a TPM is authenticated with respect to its endorsement key EK, the issuer can be sure that it is dealing with a given TPM that it has manufactured and not previously issued a credential to.

If this situation holds, then we can configure our P2P software with a list of manufacturers' public keys used for issuing credentials (much in the same way that SSL is usually configured with a list of root public keys). Alternatively, manufacturers may choose to obtain certificates for their public keys from other certification authorities, in which case some form of PKI would be involved. Here we are assuming that the number of manufacturers of TCG-enabled platforms would be a small and relatively static group. This would allow an owner of a platform created by one manufacturer to transparently verify a DAA signature from a platform assembled by a different manufacturer while maintaining a high level of confidence in the authenticity of the end platform.

The obvious question is what does this mean for P2P computing? The traditional view of P2P networking is that it is an all-you-can-eat buffet of file sharing, where content flows as fast as connection bandwidth permits. At the same time, it is a commonly held belief that Trusted Computing represents a threat to this free-for-all as a tool for enabling Digital Rights Management (DRM). However, our analysis shows that the opposite may be the case. If TCG-compliant platforms become prevalent on end user systems, then the techniques developed here will make it expensive for copyright holders or their agents to engage in pseudospoofing and the flooding of networks with poor-quality content. With the development of suitable reputation mechanisms, the reputations of the offending peers would rapidly decline and they could be barred altogether. Our techniques will also enable users of P2P networks to maintain secure end-to-end communications, immune from monitoring by Internet Service Providers or other entities. This could significantly hamper the prosecution of individuals

operating illegally on P2P networks. Using simple access control lists based on pseudonyms, small groups could set up completely private, access-controlled subgroups within a given P2P network. Users could form their own groups based on content of choice or personal preferences, and so an existing P2P network could be utilised to bootstrap multiple private P2P networks. There is naturally the danger of such a feature being abused by certain groups. It would be difficult to prevent the formation of such groups, given that any platform can act as an issuer (and not just manufacturers).

6.2 Securing Commercial P2P Networks Using Trusted Computing

There is an alternative model to the one presented above which is more commercially-orientated in application. In this model we reposition the root of trust from the platform manufacturer to a service/content provider. The security measures that we present here are not intended to provide a solution to all the possible problems that may be potentially encountered in a real world setting, but they do address some of these issues, such as discouraging bad behaviour by peers.

We will begin by sketching a relatively simple example using a fictional application herein referred to as TrustedPeer. In this example we also make use of a fictional TTP named ContentCorp. In our scenario ContentCorp provides both the TrustedPeer application as well as acting as an issuer of DAA credentials.

Peers download the application from a ContentCorp server and install it on their TCG enabled device. In order for a peer to become a member of the network, it must register with ContentCorp. DAA can be used for the registration. A successful run of the join phase provides a peer with a credential in the form of a CL signature on its secret value f (as in Section 3.4). This credential can later be used by the peer to authenticate itself to the service provider and to other peers through the DAA-Signing protocol. Thus this credential effectively becomes proof of registration.

During this registration procedure the platform must reveal a pseudonym $N_I = \zeta_I^f$. Here the quantity ζ_I is set by ContentCorp and allows them to associate a "handle" with a peer as well as to check for rogue platforms. The handle may be used later by ContentCorp, for example at the stage of collecting payment for access (in which case, it may be necessary to collect additional payment details at the time of registration). Once this stage is complete a peer can gain access to the TrustedPeer network.

So far, the approach we have described is broadly similar to a TCG application scenario described in [1]: we have a centralised infrastructure making content available to subscribing peers. The benefits this approach offers are twofold. The benefit to service providers is that only platforms that have a credential issued by them, i.e. that are registered, will be permitted access to the network and its content. It benefits registered participants insofar as other platforms will not be able to impersonate them and gain access to content that they have paid for.

This type of content distribution application could start to evolve into a more decentralised P2P architecture by allowing peers to obtain content from other peers as the content spreads out from ContentCorp to subscribing peers. In this scenario a peer looking for specific content sends out a query onto the network, receives a list of matching hits, examines them for relevance and then decides which peer to use to download the required content. In order that peers can distinguish genuine ContentCorp content from rogue content, ContentCorp could digitally sign their content and embed their public key in the TrustedPeer application.

As more and more content becomes distributed, query results for content could start to be returned by other peers on the network, thus reducing the load at ContentCorp servers.

The TrustedPeer application could be configured to demand that the requesting peer provide a proof (in the form of a CL signature) that it is a peer registered with ContentCorp before supplying the requested content. It could be further configured to supply the requesting peer with the content in an encrypted form, with the session key used for encryption further encrypted under ContentCorp's public key, perhaps along with the responding peer's pseudonym and some metadata describing the content. It would then be incumbent on the requesting peer to contact the ContentCorp server to obtain the session key, at which point ContentCorp could collect payment for the content. Here, the communication with ContentCorp would involve only short messages and low bandwidth rather than the high bandwidth required for content distribution. Further, ContentCorp would have visibility of which peers were supplying content, and those peers could perhaps be rewarded appropriately (perhaps with reduced fees for subsequent downloads). Thus ContentCorp could generate revenue from its content without being directly involved in its distribution.

This approach could be further enhanced by adding a reputation system, enabling peers to search out peers with fast connections or rich content. Building on the stable pseudonyms that are enforced by the DAA protocols, such a mechanism would naturally resist pseudospoofing and pseudotheft attacks. ContentCorp could also keep state regarding which EKs have already been registered in order to prevent multi-credentialised platforms joining the network.

One important issue that remains to be addressed is the prevention of "leakage" of content from the ContentCorp network. There is nothing in our system as described so far that prevents a peer registering, obtaining content, and then re-distributing it on traditional, anarchic P2P networks. There is a form of economic incentive to prevent this activity: a peer, having paid for the content, may be less likely to distribute it freely to others. Moreover, a peer who knows that ContentCorp will give him credit of some kind if other peers download content from him is more likely to conform and use the encrypted session key mechanism that we have described above. In turn, this forces peers wishing to obtain content to pay ContentCorp for it.

A technical measure that could further discourage unrestricted re-distribution of content would be to use watermarking, marking content with the requesting user's DAA pseudonym before distribution. In the event that a peer offers content from the TrustedPeer network on another network, then the peer could be identified via the watermark and potentially permanently banned from the TrustedPeer network. Of course, there is the problem of "watermark handoff" in the case where peers themselves become responsible for distributing content: if peer A sends content X to peer B then the watermark on content X would need to be changed from ζ^{f_A} to ζ^{f_B}.

A far stronger mechanism would be to rely on the ability of a TCG-enabled platform to lock availability of content to particular software configurations: with this in place, ContentCorp could be assured that the correctly behaving TrustedPeer application is in operation before content is available at a peer. This would limit a rogue peer's ability to re-distribute content. However, this kind of mechanism would depend on the extension of the domain of control of TCG from the OS level to applications themselves. This is certainly something we expect to see in the years ahead. (We emphasise that none of the other security techniques built from TCG that we have outlined in this paper require that kind of control.)

7 Issues and Open Problems

We briefly summarise some issues with our approach to securing P2P networks using trusted computing. We also highlight some open problems and areas suitable for further development.

7.1 Issues

7.1.1 Credential Replacement

Replacement of credentials is a non-trivial task in DAA. If it is necessary to issue replacement credentials then the previous DAA history of the platform gets erased, at least as far as the DAA credentials from that issuer are concerned. This potentially allows a rogue to rejoin a system from which it has previously been barred. However decisions on credential replacement are a matter of policy for a given issuer. For example, an issuer can consult a black list before issuing a credential to a platform.

7.1.2 Data Representation

Another issue in our approach to securing P2P networks is that of data representation. There is something of a semantic gap between dealing with a DAA pseudonym (a sequence of ones and zeros) and working with the more traditional 'handles' to which users of P2P networks are accustomed. One possible solution to this problem is to link a local name space with a network name. If a peer on a particular network maintains a list of pairs (ζ, ζ^f) then with each of these pairs it can associate a particular name or handle. In this scenario ζ represents the network and ζ^f a particular pseudonym on that network. This mapping would be akin to that used in SPKI/SDSI[10] but instead of using a public key for identification as in SPKI/SDSI, we would use (ζ, ζ^f). In SPKI/SDSI every principal maintains a name space where names are bound to values, possibly in other principal's name spaces. This kind of mechanism is suitable for use in a P2P network as it is not necessary for peers to know every other peer on a network. Instead, they need only know those peers with whom they have previously interacted or are intending to interact.

7.2 Open Problems

7.2.1 Reputation

One area of our work where further development is possible is that of reputation systems for TCG-enabled P2P networks. For example, it may be possible to set up a recommendation system whereby peers pass on their experiences to other peers by sending a subset of their named handles to other peers following a SDSI naming convention. In this case peers could provide DAA-signed reference subsets based on successful interactions with other peers. It would be interesting to see various established rating schemes examined in the context of TCG-based P2P networks: it would seem that they could be far more robust than is currently the case in standard P2P networks. Given the network based unlinkability property established in Section 4, developing the ability to securely transfer a peer's pre-existing reputation in one network to another seems like an interesting challenge.

[10] See for example `http://www.ietf.org/html.charters/spki-charter.html`

7.2.2 Confidentiality and Integrity of stored data

We have mainly concentrated on confidentiality and integrity of data *in transit* in P2P networks. A TCG-enabled platform provides *sealing* mechanisms which, in combination with attestation, can provide confidentiality and integrity for stored data. Such secure storage mechanisms could be exploited to further enhance the security of P2P networks, and it would be interesting to see further research in this area.

7.2.3 Anonymity

Anonymity challenges other security goals within P2P networks such as accountability and reputation. DAA provides a certain degree of anonymity in the form of unlinkability of transactions arising from a user, but by fixing the name base (ζ) in our approach we have restricted the provision of such anonymity. However, anonymity within multiple networks is still present as long as different name bases (ζ) are used. Providing complete anonymity would bring with it problems relating to routing and discovery of resources, but it would be worthwhile continuing research in this area.

Research into P2P based architectures has given rise to various mechanisms to achieve anonymity, for example, onion routing and MIX networks. It would be interesting to see the extent to which these mechanisms could be combined with our ideas to enhance the anonymity properties of our approach without compromising its other security properties.

7.2.4 Authenticity

Another issue within anarchic P2P networks is the question of how to guarantee the authenticity of resources. There has been considerable work done within the research community to establish certain measures of authenticity. In the P2P context, Daswani *et al.* [9] have proposed evaluating authenticity based on expert opinion, polls or first-serve basis. In our work, we have not identified authenticity as an important security issue in P2P networks, though we briefly touched on it in Section 2.2.1. Our work in Section 6.2 does provide an explicit authenticity mechanism, in the form of digital signatures on resources by the content provider. By preventing pseudospoofing attacks in anarchic P2P networks, our methods also go some way to ensuring resource authenticity in that less-restricted world. It would be interesting to explore further how resource authenticity could be enhanced using TCG in such networks.

8 Conclusions

In this paper, we have outlined the security issues faced in P2P networks and discussed the extent to which features of the TCG specifications can be used to enhance security. All but one of our proposals for enhancing security use only those TCG features which are in place and available on platforms today. We have applied our ideas in two distinct types of P2P network: one anarchic and modelling today's file sharing networks, and the other more commercially oriented. One perhaps counter-intuitive implication of our work is that TCG-enabled P2P networks can make P2P networks harder to effectively police. Our work also points the way forward to using TCG to enable secure P2P-commerce, through the development of secured commercial P2P networks and of P2P networks free from pseudospoofing and pseudotheft.

Acknowledgements

We wish to convey our thanks to Liqun Chen and Graeme Proudler of Hewlett-Packard Laboratories Bristol for answering many questions on the TCG specifications and their applications.

References

[1] J. Camenisch. Direct anonymous attestation: achieving privacy in remote authentication. Talk given at Zurich Information Security Colloquium, `http://www.zisc.ethz.ch/events/infseccolloquium2004`.

[2] J. Camenisch and A. Lysyanskaya. A signature scheme with efficient protocols. In S. Cimato, C. Galdi, and G. Persiano, editors, *Security in Communication Networks, Third International Conference, SCN 2002*, volume 2576 of *LNCS*, pages 268–289. Springer, 2003.

[3] M. Castro, P. Druschel, A. M. Kermarrec, and A. Rowstron. Scribe: a large-scale and decentralised application-level multicast infrastructure. *IEEE Journal on Selected Areas in Communications*, 20(8):1489–1499, October 2002.

[4] D. L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Communications of the ACM*, 24(2):84–88, 1981.

[5] L. Chen, E. Brickell, and J. Camenisch. Direct anonymous attestation. In V. Atluri, B. Pfitzmann, and P. McDaniel, editors, *Proceedings of 11th ACM Conference on Computer and Communications Security – CCS 2004*, October 2004.

[6] I. Clarke, O. Sandberg, B. Wiley, and T. W. Hong. Freenet: a distributed anonymous information storage and retrieval system. In H. Federrath, editor, *Proceedings of International Workshop on Design Issues in Anonymity and Unobservability*, volume 2009 of *LNCS*, pages 46–66. Springer, 2001.

[7] E. Damiani, S. De Capitani di Vimercati, S. Paraboschi, P. Samarati, and F. Violante. A reputation-based approach for choosing reliable resources in peer-to-peer networks. In V. Atluri, editor, *Proceedings of the 9th ACM conference on Computer and Communications Security*, pages 207–216. ACM Press, 2002.

[8] N. Daswani, H. Garcia-Molina, and B. Yang. Open problems in data-sharing peer-to-peer systems. In D. Calvanese, M. Lenzerini, and R. Motwani, editors, *Proceedings of 9th International Conference on Database Theory (ICDT'03)*, volume 2572 of *LNCS*, pages 1–15. Springer, 2003.

[9] N. Daswani, P. Golle, S. Marti, H. Garcia-Molina, and D. Boneh. Evaluating reputation systems for document authenticity. Technical report, Computer Science Department, Stanford University, 2003.

[10] A. Datta, M. Hauswirth, and K. Aberer. Beyond 'web of trust': enabling P2P e-commerce. In *Proceedings of the IEEE Conference on Electronic Commerce (CEC'03)*, pages 303–312. IEEE Computer Society Press, June 2003.

[11] L. Detweiler. The snakes of medusa – Internet identity subversion, 1993. `http://www.interesting-people.org/archives/interesting-people/199311/ms%g00054.html`.

[12] T. Dierks and C. Allen. Request for Comments 2246 – The TLS Protocol Version 1.0, January 1999.

[13] R. Dingledine, M. J. Freedman, and D. Molnar. Accountability. In Oram [25], pages 171–213.

[14] J. R. Douceur. The Sybil attack. In P. Druschel, M. F. Kaashoek, and A. I. T. Rowstron, editors, *Peer-to-Peer systems, First International Workshop, IPTPS 2002*, volume 2429 of *LNCS*, pages 251–256. Springer, 2002.

[15] P. Fenkam, S. Dustdar, E. Kirda, H. Gall, and G. Reif. Towards an access control system for mobile peer-to-peer collaborative environments. In *IEEE 11th International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE 2002)*, pages 95–101. IEEE Computer Society Press, June 2002.

[16] T. Garfinkel, M. Rosenblum, and D. Boneh. Flexible OS support and applications for Trusted Computing. In *9th USENIX Workshop on Hot Topics in Operating Systems (HOTOS-IX)*, pages 145–150, 2003.

[17] D. Harkins and D. Carrel. Request for Comments 2409 – The Internet Key Exchange (IKE), November 1998.

[18] International Organisation for Standardization. *Information processing systems – Open Systems Interconnection – Basic Reference Model – Part 2: Security Architecture (ISO 7498-2)*, 1989.

[19] S. Iyer, A. Rowstron, and P. Druschel. Squirrel: a decentralized peer-to-peer web cache. In *Proceedings of the twenty-first annual symposium on Principles of distributed computing*, pages 213–222. ACM Press, 2002.

[20] M. Kinateder and S. Pearson. A privacy-enhanced peer-to-peer reputation system. In K. Bauknecht, A. M. Tjoa, and G. Quirchmayr, editors, *EC-Web 2003*, volume 2738 of *LNCS*, pages 206–216. Springer, 2003.

[21] J. Kubiatowicz, D. Bindel, Y. Chen, S. Czerwinski, P. Eaton, D. Geels, R. Gummadi, S. Rhea, H. Weatherspoon, W. Weimer, C. Wells, and B. Zhao. Oceanstore: an architecture for global-scale persistent storage. In *Proceedings of the Ninth international Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS 2000)*, pages 190–201. ACM Press, November 2000.

[22] S. Marti and H. Garcia-Molina. Identity crisis: anonymity vs. reputation in P2P systems. In *Proceedings of the 3rd International Conference on Peer-to-Peer Computing*, pages 134–141. IEEE Computer Society Press, September 2003.

[23] D. S. Milojicic, V. Kalogeraki, R. Lukose, K. Nagaraja, J. Pruyne, B. Richard, S. Rollins, and Z. Xu. Peer-to-peer computing. Technical Report HPL-2002-57, HP Labs, March 2002. `http://www.hpl.hp.com/techreports/2002/HPL-2002-57.html`.

[24] D. Molnar, R. Dingledine, and M. J. Freedman. Free haven. In Oram [25], pages 102–120.

[25] A. Oram, editor. *Peer-to-peer: harnessing the power of disruptive technologies*. O'Reilly & Associates, Inc., 2001.

[26] M. K. Reiter and A. D. Rubin. Crowds: anonymity for web transactions. *ACM Transactions on Information and System Security*, 1(1):66–92, 1998.

[27] M. Rennhard and B. Plattner. Practical anonymity for the masses with Mix-networks. In *Proceedings of the IEEE 8th International Workshop on Enterprise Security (WET-ICE'03)*, pages 255–260. IEEE Computer Society Press, June 2003.

[28] P. Resnick, K. Kuwabara, R. Zeckhauser, and E. Friedman. Reputation systems. *Communications of the ACM*, 43(12):45–48, 2000.

[29] S. E. Schechter, R. A. Greenstadt, and M. D. Smith. Trusted computing, peer-to-peer distribution and the economics of pirated entertainment. In *The Second Workshop on Economics and Information Security*, May 2003.

[30] P. F. Syverson, D. M. Goldschlag, and M. G. Reed. Anonymous connections and onion routing. In *Proceedings of IEEE Symposium on Security and Privacy*, pages 44–54. IEEE Press, 1997.

[31] Trusted Computing Group. TCG software stack specificiation version 1.1, 2003. `https://www.trustedcomputinggroup.org/downloads/specifications`.

[32] Trusted Computing Group. TPM main: Part 1 design principles, 2003. `https://www.trustedcomputinggroup.org/downloads/specifications`.

[33] Trusted Computing Group. TPM main: Part 3 commands, 2003. `https://www.trustedcomputinggroup.org/downloads/specifications`.

[34] Trusted Computing Group. TCG specification architecture overview revision 1.2, 2004. `https://www.trustedcomputinggroup.org/downloads/specifications`.

[35] D. S. Wallach. A survey of peer-to-peer security issues. In M. Okada, B. C. Pierce, A. Scedrov, H. Tokuda, and A. Yonezawa, editors, *Software Security – Theories and Systems, International Symposium, ISSS 2002*, volume 2609 of *LNCS*, pages 42–57. Springer, 2003.

[36] P. Zimmermann. *PGP source code and internals*. MIT Press, 1995.