A decentralized public key infrastructure for customer-to-customer e-commerce*

Karl Aberer, Anwitaman Datta, and Manfred Hauswirth
Distributed Information System Lab, Swiss Federal Institute of Technology, Lausanne (EPFL), CH-1015 Lausanne, Switzerland
{karl.aberer, anwitaman.datta, manfred.hauswirth}@epfl.ch

Abstract: The success story of eBay has shown the demand for customer-to-customer (C2C) electronic commerce. eBay is a centralized infrastructure with all its scalability problems (network bandwidth, server load, availability, etc.). In this paper we argue that C2C e-commerce is an application domain that maps naturally onto the emerging field of peer-to-peer (P2P) systems simply by its underlying interaction model of customers, i.e., peers. This offers the opportunity to take P2P systems beyond mere file sharing systems into interesting new application domains. The long-term goal would be to design a fully functional decentralized system which resembles eBay without eBay's dedicated, centralized infrastructure. Since security (authenticity, non-repudiation, trust, etc.) is key to any e-commerce infrastructure, our envisioned P2P e-commerce platform has to address these security issues adequately. As the first step in this direction we present an approach for a completely decentralized P2P public key infrastructure (PKI) which can serve as the basis for higher-level security service. We base it on a statistical approach and present an analytical model to quantify its behavior and properties and to provide probabilistic guarantees. To justify our claims we provide a first-order analysis and discuss the PKI's resilience against various known threats and attack scenarios.

Keywords: Customer-to-customer e-commerce, public key infrastructure, peer-to-peer systems.

Biographical notes:

Karl Aberer is a full professor at EPFL since September 2000. There he is heading the Distributed Information Systems Laboratory of the School of Computer and Communications Sciences. His main research interests are on distributed information management, P2P computing, semantic web and the self-organization of information systems. He received his Ph.D. in mathematics in 1991 from the ETH Zürich. From 1991 to 1992 he was postdoctoral fellow at the International Computer Science Institute (ICSI) at the University of California, Berkeley. In 1992 he joined the Integrated Publication and Information Systems institute (IPSI) of GMD in Germany, where he became manager of the research division Open Adaptive Information Management Systems in 1996. He has published more than 80 papers on data management on the WWW, database interoperability, query processing, workflow systems and P2P data management. Recently he has been PC chair of ICDE 2005, DBISP2P 2003, RIDE 2001, DS-9, and ODBASE 2002. He is associate editor of SIGMOD RECORD and member of the editorial board of the VLDB Journal and Web Intelligence and Agent Systems.

Anwitaman Datta is a research assistant from the Distributed Information Systems Laboratory at the Swiss Federal Institute of Technology in Lausanne (EPFL). He holds a B.Tech. in Electrical Engineering from the Indian Institute of Technology (IIT Kanpur). Prior to joining EPFL, he worked at Transwitch India as an associate member of technical staff. Anwitaman is currently pursuing a Ph.D. under the supervision of Prof. Karl Aberer. His research interests include peer-to-peer systems, ad-hoc networks, epidemic algorithms and self-organization and resilience of decentralized systems. He is a member of the

* The work presented in this paper was supported (in part) by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS).

Manfred Hauswirth is a senior researcher at the Distributed Information Systems Laboratory of the Swiss Federal Institute of Technology in Lausanne (EPFL) since January 2002. He holds an M.S. (1994) and a Ph.D. (1999) in computer science from the Technical University of Vienna. Prior to his work at EPFL he was an assistant professor at the Distributed Systems Group at the TU Vienna where he still lectures courses on distributed systems. His main research interests are on peer-to-peer systems, distributed information management, self-organizing systems, semantic web, and security in distributed systems. He has published numerous papers and co-authored a book on distributed software architectures. He has served in numerous program committees of international scientific conferences and is a member of IEEE and ACM.

INTRODUCTION

The demand for customer-to-customer commerce (C2C) has been proven by the huge success of eBay. eBay provides a centralized trading platform to its customers and offers a certain degree of security that business transactions between partners, which are unknown to each other, are performed properly. The advantage of eBay's centralized architecture is that rules can be enforced easily. However, this turns into a severe problem if we switch the viewpoint: In any centralized architecture the central entity is a single point of failure and a bottleneck in terms of bandwidth and computing resources which limits scalability and in turn causes high infrastructure requirements.

For these reasons we argue that a customer-to-customer system would map more naturally onto a P2P system by its very structure and interaction patterns. However, more effort has to be put into the design in order to offer similar services and guarantees as a centralized infrastructure. There may be additional scepticism about the viability of trading high-valued commodities in a P2P commerce platform. However, even with centralized solutions, the risks are almost equally high that the other customer does not deliver what he promises and thus C2C commerce will indeed be more popular for trading commodities of relatively less value.

While scalability and fault-tolerance come implicitly with P2P infrastructures, as has been proven by successful P2P systems such as Kazaa or Gnutella, security guarantees similar to centralized architectures are more difficult to achieve in a decentralized environment. Many security services, for example, authentication of the trading partners, rely on the existence of a public key infrastructure (PKI). PKIs typically have centralized architectures which contradicts the P2P approach and would introduce centralization into a decentralized approach again through the back door and would defeat the advantages gained from decentralization. An alternative would be to use a PGP-like "web of trust"' approach. However, it has been shown that this concept has several severe shortcomings as we will discuss in the next section. Thus we suggest a different strategy and propose a decentralized PKI based on a statistical (quorum-based) approach that overcomes these problems.

To set the stage we start with a classification and discussion of existing PKI approaches. Then we present a detailed description of the decentralized PKI architecture we propose by discussing all its building blocks and algorithms. To justify the validity of our model we then provide a first-order analysis of the incurred effort and security properties of our PKI. We continue with a security analysis of our PKI in which we analyze common attacks with respect to our approach. This is followed by a discussion of related work, which is relatively sparse in this domain, due to the pioneering character of our work and we end with our conclusions.

PKI APPROACHES

This section defines an informal taxonomy to classify existing PKI approaches, discusses the main variants, and positions our proposed system with respect to this classification.

Taxonomy of PKI approaches

In the context of data management, current PKIs may be classified either as centralized or decentralized. A centralized PKI consists of a federation of trusted third parties (TTP), so-called certification authorities (CA), for example, VeriSign. The TTPs do not participate in the interactions of a system but act as facilitators of the activities. We omit these centralized solutions from the rest of our discussion since they are not in our scope and have already been evaluated in great detail, for example, in [16].

In decentralized PKIs the public key infrastructure is maintained by the participants themselves without using central control and special roles such as CAs. Essentially decentralized PKIs can be categorized by the strategy that is applied to discover peers which store the public key of an entity and how much these peers can be trusted. Three main subclasses of decentralized PKIs exist: web-of-trust, statistical (quorum-based) approaches, and hybrid approaches.

In the *web-of-trust model* a participant of the system knows the public keys of some other peers, and considers this knowledge sacrosanct. It also relies on some of these peers (with a possibly varying degree of trust) to certify the

public key of other peers. Then to obtain a trustworthy public key of another peer a path in the peer acquaintance (trust) graph has to be found. This graph, the so-called *web-of-trust*, is formed by a simple transitive trust rule: If P_A trusts that K_B is P_B's public key, and also relies (personally determined) on P_B to certify a third party's public key, then P_A will believe that K_C is P_C's public key if P_B certifies it. PGP [12] and its variants belong to this group. PGP implicitly exploits the small world phenomenon of social acquaintance that is observed in the trust (certificate) graph [5] to create a web-of-trust of peers' public keys. Consequently, it obliterates the need of central authorities, and has been enormously successful as a freely available decentralized public key infrastructure. However, the strength of a trust chain is determined by its weakest link, and hence a simple transitive trust model as PGP's is highly vulnerable and thus unreliable.

Statistical (quorum-based) approaches obtain the public key information from many peers and if it is possible to form a quorum the public key is considered authentic. We will elaborate on this type of PKIs in the next section where we describe our approach. The essential idea behind a statistical approach is to have multiple random and thus presumably independent sources (peers) which replicate the public key information, and to retrieve the information from a random subset of these replicas. This requires an efficient, decentralized indexing infrastructure. The approach presented in this paper belongs to the group of statistical approaches and we use our P2P storage management system P-Grid [4] for efficient indexing and retrieval of public key information. However, it should be emphasized that any other P2P system with guarantees similar to the ones of P-Grid may be used, and thus our proposal is generic and could be adapted to other decentralized overlay networks.

In *hybrid approaches* public key information is obtained from many, preferably independent peers, and then a weighted quorum is formed depending on the relative trust level of the information providing peers.

Discussion of decentralized PKI approaches

As stated above the two dimensions relevant to decentralized PKIs are the strategy to discover peers which have the required public key and the trustworthiness of the information (public key) obtained from these peers. A fundamental difference between the web-of-trust and the statistical approaches is the strategy in which the two dimensions are navigated, i.e., the mechanism by which the relevant information is obtained: random walks in a trust graph or systematic, efficient access to relevant information, and then using a quorum (possibly weighted) for reliability (trust).

The inefficiency in web-of-trust based approaches arises primarily from the fact that information is searched through random walks. While random walks are the best (though inefficient) strategy to discover information in structure-less P2P systems such as Gnutella, structured P2P systems support efficient searches, inserts, and updates which can be exploited to build statistical or hybrid PKIs efficiently. Another reason in favour of using structured P2P systems as a base technology is that a quorum-based approach in structure-less P2P systems is impractical because searches depend on flooding (and thus are inefficient), and also maintenance (inserts or updates) has high overhead.

Apart from the inefficiency of random walks, web-of-trust based approaches have further drawbacks:

1. Path discovery is inefficient because the effort is not shared, and has high, unbounded latency.

2. Web-of-trust approaches have primarily been applied for privacy purposes. To completely believe in a person's public key, it has to be obtained offline or the public keys must have been certified by a known and trusted entity (person). It is thus premature to assume, particularly in the absence of any quantifiable guarantees, that the web-of-trust model that worked well for email privacy can be converted into an all-purpose decentralized public key infrastructure.

3. Web-of-trust models fail to use the collective knowledge of the whole population, but use only information available within a small number of transitive hops, determined by time to live, in the connectivity graph. However, the ultimate purpose of a decentralized PKI should be to provide a way to establish identity beyond reasonable doubt. Web-of-trust models cannot guarantee this because transitive paths are not guaranteed.

4. Finally, since the trust on other peers' certification is essentially ad-hoc, the web-of-trust is susceptible to the malicious behaviour of a few or even a single peer.

An extension of PGP's original approach is to include multiple paths of trust transitivity [15] to improve the reliability of authentication. Nevertheless, the reliability of such approaches is limited because of intersecting paths, which requires authentication metrics [15] to quantify the reliability of such multiple paths. Thus it consumes a lot of network and computational resources, worsened by the fact that there are no guarantees for the existence of multiple independent paths, nor any mechanism to discover them efficiently exists. This approach again is an attempt to navigate the two dimensions mentioned at the beginning of this section in the wrong order, since random walks are used for information discovery with the assumption of finding multiple paths, trying to exploit the power-law distribution of trust graphs. A further weakness of this approach is that the multiple paths and the metrics need to be evaluated at each peer, and thus the effort is not shared.

Based on the above discussions we argue that statistical and hybrid approaches are preferable over web-of-trust

approaches. Their efficient implementation which was a major problem for a long time due to the lack of adequate infrastructures for reliable, distributed information maintenance is now facilitated by structured P2P systems. The application of structured P2P system for PKIs based on statistical or hybrid approaches avoids the inefficiencies of the web-of-trust model and additionally provides quantifiable probabilistic guarantees which are not available for web-of-trust approaches.

A QUORUM-BASED DECENTRALIZED PKI

A prerequisite for decentralized PKIs is an efficient and reliable distributed information access structure that facilitates efficient updates and replication, tolerates frequently disconnected peers and accounts for uncooperative peers. This is not possible in unstructured P2P systems like Gnutella. Thus the so far pre-dominant "web-of-trust" model employs random walks (basically flooding) for exploring the trust graph of a P2P network. But with the recent development of efficient access structures such as CAN, Chord, Freenet, or P-Grid, it is indeed possible to realize more systematic models, rather than relying on the ad-hoc "web-of-trust" model, for which no probabilistic guarantees are available so far. Our approach to decentralized PKI is based on P-Grid. Thus we start with a brief introduction of P-Grid [4], before elaborating the details of our PKI approach.

P-Grid

P-Grid is a distributed data structure based on the principles of distributed hash tables (DHT) [13]. As any DHT approach P-Grid is based on the idea of associating peers with data keys from a key space $\mathcal{K}$. Without constraining general applicability we will only consider binary keys in the following.

Each peer $p \in Peers$ in P-Grid is associated with a binary key from $\mathcal{K}$. We denote this key by *path(p)* and call it the path of the peer. This key determines which data keys the peer has to manage, i.e., the keys in $\mathcal{K}$ that have *path(p)* as prefix. In order to ensure that the complete search space is covered by peers we require that the set of peers' paths is *complete*, i.e., for every prefix s_{pre} of the path of a peer p there exists a peer p' such that $path(p')=s_{pre}$ or there exist peers p_0 and p_1 such that $s_{pre}0$ is a prefix of $path(p_0)$ and $s_{pre}1$ is a prefix of $path(p_1)$. Naturally, one of the two peers p_0 and p_1 will be p itself in that case. Put differently, a P-Grid is constructed as a distributed binary tree. Completeness is guaranteed by P-Grid's construction algorithm. Additionally the construction algorithm ensures that a P-Grid will converge to a state where it is *prefix-free* for optimal load balancing, i.e., for peers p_0 and p_1 we have $\neg (path(p_0) \subseteq path(p_1) \vee path(p_1) \subseteq path(p_0))$, where $s \subseteq s'$ denotes the prefix relationship among strings s and s'. Algorithms for construction and maintenance and for load-balancing of a P-Grid have been introduced in [2].

To provide robustness in the presence of peer failures and distributed search workload, each path is covered by multiple peers (replicas). For searches each peer maintains *routing tables* which are defined as (partial) functions $ref: Peers \times N \rightarrow \{Peers\}$ with the properties

1. $ref(p,l)$ is defined for all $p \in Peers$ and $l \in N$ with $1 \leq l \leq |path(p)|$
2. $ref(p,l) \subseteq Peers_{s1s2...s\{l-1(1-sl)}$ with $path(p)=s_1s_2...s_{l-1}$ $s_l...s_k$, $k \geq l$

where $Peers_t = \{ p \in Peers \mid t \subseteq path(p)\}$ for $t \in \mathcal{K}$.

Having multiple references at each level l again is necessary to guarantee robustness of the data structure. We denote by r the maximum number of references maintained at each level. The search algorithm for locating data keys indexed by a P-Grid is defined as follows: Each peer $p \in Peers$ is associated with a location *loc(p)* (IP address). Searches can start at any peer. Peer p knows the locations of the peers referenced by *ref(p,l)*, but not of other peers. Thus the function *ref(p,l)* provides the necessary routing information to forward search requests to other peers in case the searched key does not match the peer identifier. Let $t \in \mathcal{K}$ be the searched data key and let the search start at $p \in P$. Then search is defined by the following recursive algorithm:

Algorithm 1 Search in P-Grid: search(t, loc(p))

```
1: if path(p) ⊆ t then
2:     return(loc(p));
3: else
4:     determine maximal l such that t_1...t_{l-1}(1-t_l) ⊆ path(p);
5:     r = randomly selected element from ref(p,l);
6:     return (search(t, loc(r)));
7: end if
```

Due to the definition of *ref*, the function *search* will always find the location of a peer at which the search can continue (use of completeness). With each invocation of *search(t, loc(p))* the length of the common prefix of *path(p)* and t increases at least by one. Therefore the algorithm always terminates. The average search cost irrespective of the data distribution is $O(\log n)$ with n being the number of leaves in the P-Grid tree.

Unlike most contemporary P2P systems P-Grid supports updates of the stored, replicated data using a push/pull gossiping mechanism which provides probabilistic success and consistency guarantees even in unreliable network environments [8].

A decentralized PKI based on P-Grid

Each peer p is uniquely identified by a universally unique identifier (UUID) Id_p which is generated once when a peer joins the P-Grid community, by applying a cryptographically secure hash function to some random data. Routing tables hold only these identifiers. Each peer p additionally has a cache that holds mappings of peers' Id_p to their physical addresses $addr_p$ it already knows ((Id_i, $addr_i$, TS_i), where TS_i denotes a time-stamp) in order to be able to communicate with other peers. Since disconnections of peers may lead to changing IP addresses, correct mappings

need to be maintained which is provided inside P-Grid itself as described in [1] using P-Grid's update functionality [8]. At bootstrap, each peer p also generates a private/public key pair D_p/E_p. Then our algorithm for building a decentralized PKI on top of P-Grid works as given below.

In the following discussion, $R_{min4} \leq R_{min3} \leq R_{min2} \leq R_{min1}$, $T_{out2} > T_{out1}$ are design parameters determining the level of security and influencing the performance. We will use these for our analysis in the next section of the effort required and performance (probabilistic) offered by our system. For simplicity we will consider $R_{min4} = R_{min3}$ without loss of generality of the results of our analysis.

Bootstrap (a new peer *p* joins the P-Grid)

1. p determines its current IP address and generates Id_p, D_p/E_p.

2. (Id_p, $addr_p$, E_p, TS_p, $D_p(Id_p, addr_p, E_p, TS_p)$) (for brevity denoted as *tuple* in the following) is inserted into P-Grid by p using Id_p as the key (TS_p is a timestamp used to prevent replay attacks, $D_p(x)$ means that the request is signed by p). The Id/IP mapping is inserted in the P-Grid to be able to handle possible changes of physical addresses and has been described and analyzed in [9]. Inserting in P-Grid means that the request is routed to a responsible replica peer, $p_i \in \Re_p$. $\Re_p$ is the set of replicas responsible for storing information about peer p. If Id_p already exists in the P-Grid (though this is very improbable) p generates a new Id_p and repeats this step.

3. p repeats the insertion operation at R_{min1} random and distinct P-Grid peers, so that the insertion request reaches an expected R_{min2} distinct replicas.

4. All p_i that receive the *insert(tuple)* message initiate *update(tuple,p_i)* among their replicas $\Re_p$. All replicas, including the ones that originated such updates locally store the tuple only if they receive and can form a quorum of $R_{min3} \leq R_{min2}$ distinct update messages within T_{out1} time. Peers who received the original insert then send a confirmation to p. This of course holds for the peers/replicas that are online during the update operation. Those peers that come online later use a quorum-based pull strategy to get an up-to-date view [8]. If after T_{out1} since receiving the first update message, R_{min3} distinct messages have not been received, the peers discard the information. In the absence of Byzantine/malicious peers it would have been sufficient to make a single insert in P-Grid, since the update mechanism would have updated all replicas. However, malicious peers may initiate updates with false information. Since search and insert requests are routed to random replicas, we use multiple requests and then a quorum to address this properly.

5. As a result of the previous steps the *tuple* will be physically stored at peers in $\Re_p$. Based on the randomized algorithms that P-Grid uses we can assume that the individual replicas $p_i \in \Re_p$ are independent and they collude or behave Byzantine only to a degree that can be handled by existing algorithms.

6. If p receives $R_{min4} \leq R_{min3}$ confirmations (within some $T_{out2} > T_{out1}$), it is convinced (probabilistically) that its public key has been replicated amply for fault tolerance. Otherwise p generates a new Id_p and retries the insert process. Since only a new peer entering the P2P system needs to conduct the bootstrap phase, it is irrelevant which identity is successfully inserted. Also, generation and re-insertion of a new identity will be required only in the event of a distributed denial of service attack by malicious peers, more on which is discussed in a later section.

Peer startup (a peer *p* rejoins)

p starts up and checks whether its $addr_p$ has changed. If yes, it initiates an update of its new physical address (signed with its private key). The complete algorithm for update along with the cost incurred and its reliability can be found in [1, 9]. This step is necessary in order to make sure that the routing tables are correct.

Operation phase

This phase denotes the standard operation, i.e., p is up and running, has registered an up-to-date mapping of its identity/physical address (Id_p, $addr_p$, TS_p) and is ready to process query and update requests, for example, for other peers' public keys, physical address, or other additional information that may be present in the P-Grid. Both queries and updates need to be routed to at least one replica peer responsible for the concerned key space.

1. p receives a request Q from a peer q.

2. In case p can satisfy Q the result is returned to q. Otherwise p finds out which peer p_f to forward the query to and retrieves (Id_{pf}, $addr_{pf}$, E_{pf}, TS_{pf}) for this purpose.

3. p generates a random number ρ, contacts p_f and sends $E_{pf}(\rho)$. As an answer p_f must send ($D_{pf}(E_{pf}(\rho))$) and p can check whether $D_{pf}(E_{pf}(\rho)) = \rho$. If yes, p_f is correctly identified, i.e., p really talks to the peer it intends to, and Q is forwarded

to p_f, which means that the processing of Q at peer p was completed successfully. Otherwise the following steps are performed.

4. If step 3 failed, then p_f may have a new IP address or be offline (the case that somebody tries to impersonate p_f is covered implicitly by the signature check above) and p may either use an alternative routing reference (P-Grid routing tables hold multiple entries for each level) or send a query to P-Grid to retrieve the current $addr_{pf}$ using Id_{pf} as the key (the detailed strategy is presented and analyzed in [1].

5. p collects all answers $t_i = (Id_{pf}, E_{pf}, addr_{pf}, TS_{pf}, D_{pf}(Id_{pf}, E_{pf}, addr_{pf}, TS_{pf}))$ it receives from the $p_j \in \Re_{pf}$. If extended security is required then the p_j should sign their answers, i.e., send $(t_i, D_{pj}(t_i)))$. p has to collect at least R_{min3} answers to detect misinformed or malicious peers, i.e., checks whether a certain quorum of the answers is identical (R_{min3} is defined by each individual p according to its local requirements for trustworthiness). Otherwise the query is repeated a certain number of times before aborting. Forming a quorum can be avoided if peers store the public keys E_{pf} of other peers and decide to trust in them after they have done a certain number of (quorum-based) queries for E_{pf} that have resulted in identical answers. If p already knows E_{pf} and trusts it then it can immediately check the validity of the answer by $E_{pf}(D_{pf}(Id_{pf}, E_{pf}, addr_{pf}, TS_{pf})).Id_{pf} = t_i.Id_{pf}$.

6. Now p can proceed with step 3. In case this is successful p enters $(Id_{pf}, addr_{pf}, E_{pf}, TS_{pf}))$ in its local cache.

Any information, most importantly public keys in our context, may be accessed and maintained in the way described above. The approach of a decentralized PKI based on P-Grid is efficient because the basic operations such as search or insert in P-Grid take $O(\log\ n)$ messages to discover one random replica responsible for the relevant key. Since the routing process is randomized, attacks are hard to conduct and the reliability of results can be established by quorum-based techniques.

ANALYSIS

The costs of locating public key information reliably are a key factor for the usability of our approach which we analyze below, i.e., the costs to locate R_{min2} distinct replicas responsible for the particular key, and forming a quorum of at least R_{min3}. In the analysis we use the following notations: $|\Re_p|$ denotes the total number of replicas and R_{on} of these are online with their correct physical address known to other peers who refer to them. Note that it can be any R_{on} of the $|\Re_p|$ replicas, since the update mechanism [8] provides probabilistic consistency guarantee for the online replicas. All requests need to be routed to any one of these responsible and online replicas. The effort to route an individual request successfully in the presence of stale caches and unavailable peers has been analyzed and shown to be efficient in [9] and [1] and is logarithmic. Update costs in P-Grid with probabilistic success and consistency guarantees in the presence of peer disconnections at a reasonable overhead have been analyzed in [8] and are linear in the number of replicas. Thus we only need to analyze the expected number of independent requests R_{min1} in order to reach R_{min2} distinct replicas to form a R_{min3} quorum, where $R_{min3} \leq R_{min2} \leq R_{min1}$, out of R_{on} online replicas, assuming that $R_{on} \geq R_{min2}$.

For the analysis, we assume that the network topology is relatively static, such that R_{on} does not change dramatically during a single query propagation. This is a realistic assumption because queries in P-Grid require $O(\log\ n)$ messages, i.e., require a very short period of time. Thus the network topology may indeed be assumed to be quasi-static. Further, apart from replicas going off-line, replicas come online as well, thus the variation of R_{on} within a small period of time can be neglected for a first order analysis. Under these assumptions, we need to determine the expected number of requests at random peers in P-Grid, such that responses are obtained from R_{min2} distinct replicas out of the R_{on} online replicas. This problem may be reduced to coupon collector's problem [11], such that R_{min2} distinct coupons need to be collected out of R_{on} possible coupons. Under the assumption that all the online replicas are equally likely to be reached, which is guaranteed by P-Grid's randomized load balancing construction and routing algorithms, the expected number of requests R_{min1} is then a function of R_{on} and R_{min2}, represented as $R_{min1}(R_{on},R_{min2})$ required for R_{min2} distinct responses, and can be formalized as [11]:

$$R_{min1}(R_{on}, R_{min2}) = R_{on}(HarmonicNumber(R_{on}) - HarmonicNumber(R_{on} - R_{min2}))$$

Figure 1 shows the total expected number of P-Grid requests R_{min1} required to reach R_{min2} distinct replicas out of R_{on} online replicas (for $R_{on} = 20$ and R_{min2} is varied between 0 and 20).

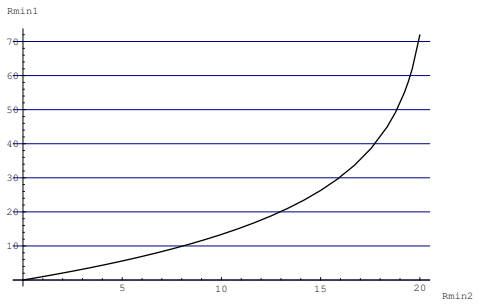

Figure 1: Expected effort (R_{min1}) for contacting R_{min2} distinct replicas among 20 online replicas

Given that a peer is malicious with probability m, the probability of successfully forming a quorum, i.e., at least R_{min3} correct replies are obtained is

$$P_{CorrectQuorum} = \sum_{i=R_{\min 3}}^{R_{\min 2}} \binom{R_{\min 2}}{i} (1-m)^i \, m^{R_{\min 2}-i}$$

The probability of a successful distributed denial-of-service (DDOS) attack, i.e., an R_{min3} quorum could not be formed, assuming that malicious peers act independently and each returns false information or does not reply at all, or malicious peers successfully persuade an enquirer with a false information (with all malicious peers collaborating and thus replying the same false information), then is

$$P_{Attack} = 1 - P_{CorrectQuorum} = \sum_{i=0}^{R_{\min 3}-1} \binom{R_{\min 2}}{i} (1-m)^i \, m^{R_{\min 2}-i}$$

Since peers are assigned a key space randomly in P-Grid, it is unlikely that all malicious replicas collaborate, and thus DDOS attacks are more probable.

Figure 2 shows the probability of successfully forming a quorum of at least 11 matching replies where 20 replicas are online, and thus can possibly be contacted under varying percentages of malicious peers.

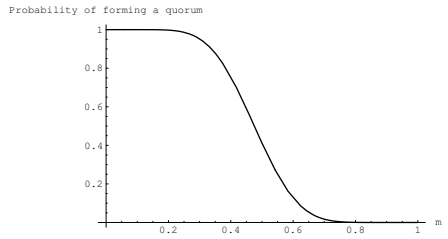

Figure 2: Probability of successfully forming a quorum with varying

The probability of a successful DDOS attack is complementary to the probability of forming a quorum. As can be seen in the figure, there is a phase transition, such that for a low percentage of malicious peers (in this example, if $m \leq 0.25$), the probability of successfully forming a quorum is close to one, and with increasing percentage of malicious peers, performance degrades rapidly and susceptibility to attacks increases. This example demonstrates that our mechanism provides quantifiable performance figures for providing authentication beyond reasonable doubt in a predominantly well-behaved P2P society.

POSSIBLE ATTACKS

Bootstrap phase

The most common attack during the bootstrap phase will be that a malicious replica inserts a wrong tuple locally instead of inserting the correct tuple, and sends an update message using this false tuple. By requiring a quorum at all well-behaved peers (replicas) we ensure that in a predominantly well-behaved P2P society, the correct information is stored.

If the percentage of malicious users is higher well-behaved peers may fail to form a quorum. This can be circumvented by repeating the insert operation with a newly generated Id_p. Since routing is randomized, it is likely that several repeated attempts will lead to eventual success.

The worst case will be that a group of collaborating malicious users are able to successfully insert a wrong public key into the P-Grid on behalf of a peer p's identity. In this case, the peer p actually has nothing to loose (since it is new, it has no identity or reputation yet), and will simply have to abandon that identity, and restart with a new one, or maybe, it will be unable to join the system. Philosophically speaking, it is not a bad option either, since most members of the system have to be malicious and collaborating in this case, so joining may not be a good idea anyway.

Peer startup phase

Here a peer p has successfully registered its correct public key in P-Grid. When p rejoins the network, it has to communicate with other peers, and p's identity will be authenticated using its public key. Here, a possible attack can be done if a peer p_v queries P-Grid to verify p's public key. Malicious peers may provide false information, thus trying to deny service. But with multiple queries starting at random peers (e.g., from p_v's routing table), this attack can be thwarted, particularly because queries are routed randomly to different replicas. Thus a quorum-based authentication will again work in a predominantly well-behaved P2P society.

Operational phase

During the operational phase, a peer p needs to first authenticate the identity of another peer p_c by conducting queries for p_c's public key. After authenticating the identity, queries related to reputation information can again be made in a similar fashion. Also, after concluding business, digital signatures may be used as proof, thus providing non-repudiation. Apart from attacks during insert or query operations, as discussed above, impersonation attacks may also be attempted.

Impersonation is prevented using public key based authentication, but depends on the percentage of malicious peers in the system. If a significant population of peers are malicious collaborate, impersonation can not be ruled out (see previous section).

RELATED WORK

Most of the work done in the context of decentralized PKIs has used PGP-like [12] web-of-trust models, trying to exploit small-world certificate graphs [5], some of which use computationally intensive authentication metrics [15]. We elaborated the drawbacks of such approaches already in detail in the section on PKI approaches and advocated the use of a statistical/hybrid model based on structured P2P

systems instead. To the best of our knowledge we have pioneered decentralized PKIs which employ statistic/hybrid approaches on the basis of structured P2P systems.

Our proposed decentralized PKI is generic, and in principle our approach could also work on top of other structured P2P systems. However, careful analysis is required for each of these systems to assess their suitability. For example, our approach would not work with the existing versions of CAN [14] or Chord [17] because they do not address the issue of updates, which is essential for our PKI approach. Freenet [7] may serve as a possible platform but would again require a detailed analysis whether it can provide sufficient guarantees especially with respect to update propagation to cached copies of information.

CONCLUSIONS

In this paper we advocated the need of a distributed PKI for P2P systems and presented our approach based on the efficient search mechanism provided by our structured P2P system P-Grid. In unstructured P2P systems like Gnutella, web-of-trust is the only available option, which is inherently inefficient and ad-hoc, probabilistic guarantees cannot be provided and costs are difficult to estimate. In contrast we employ the statistical PKI approach which facilitates sharing of effort, has low latency and guaranteed results, and provides mathematically provable probabilistic guarantees. We also provided arguments proving that our approach is resistant to various kinds of attacks.

Despite several advantages of the statistical approach, the fact remains that the web-of-trust approach is already in wide-spread use and hence future systems will possibly employ a hybrid web-of-trust/statistical approach. In fact, the routing in P-Grid itself may be thought to be very similar to traversal in trust graphs but more efficient. Incorporating web-of-trust in our approach is simple by weighting routing references according to trust in other peers, by using approaches such as [3] or [6]. Thus our work may also serve as the basis for more efficient web-of-trust approaches based on structured P2P systems. We discuss resistance of our scheme to several possible attacks, but real life experiences are still needed to identify other potential attacks and make our PKI resilient to such attacks.

Our work on identity management solely by the participating peers rather than relying on trusted third parties is a step towards enabling e-commerce in a totally P2P way. It paves way to support other services like reputation management apart from ensuring reliable functioning of the P2P system itself. For example, our approach would integrate well with work in the context of decentralized trust management, such as [3] and [6], which often require an extrinsic mechanism for authentication, but do not explicitly address this issue.

Also other domains may benefit from our approach, for example, the emerging area of web services: Each web service may be considered as an “entity” or “peer” which cooperate in a P2P way. Then service discovery [10], as well as keeping track of their quality and integrity can be achieved in a completely decentralized manner by our approach, enabling reliable inter-operation without the need of heavy-weight infrastructures.

REFERENCES

1 K. Aberer, A. Datta, and M. Hauswirth. Route maintenance overheads in DHT overlays. Technical Report IC/2003/67, EPFL, 2003.

2 K. Aberer, A. Datta, and M. Hauswirth. The Quest for Balancing Peer Load in Structured Peer-to-Peer Systems. Technical Report IC/2003/32, EPFL, 2003.

3 K. Aberer and Z. Despotovic. Managing Trust in a Peer-2-Peer Information System. In 10th International Conference on Information and Knowledge Management (ACM CIKM), 2001.

4 K. Aberer, M. Hauswirth, M. Punceva, and R.Schmidt. Improving Data Access in P2P Systems. IEEE Internet Computing, 6(1), 2002.

5 L. Buttyan, S. Capkun, and J. P. Hubaux. Small Worlds in Security Systems: an Analysis of the PGP Certificate Graph. In ACM New Security Paradigms Workshop, 2002.

6 R. Chen and W. Yeager. Poblano - A Distributed Trust Model for Peer-to-Peer Networks. http://security.jxta.org/.

7 I. Clarke, S.G. Miller, T.W. Hong, O. Sand berg, and B. Wiley. Protecting Free Expression Online with Freenet. IEEE Internet Computing, 6(1), 2002.

8 A. Datta, M. Hauswirth, and K. Aberer. Updates in Highly Unreliable, Replicated Peer-to-Peer Systems. In 23rd International Conference on Distributed Computing Systems, 2003.

9 M. Hauswirth, A. Datta, and K. Aberer. Handling Identity in Peer-to-Peer Systems. In International Workshop on Mobility in Databases and Distributed Systems (in conjunction with DEXA 2003), 2003.

10 W. Hoschek. A Unified Peer-to-Peer Database Framework and its Application for Scalable Service Discovery. In IEEE/ACM Workshop on Grid Computing (Grid'2002), 2002.

11 R. Motwani and P. Raghavan. Randomized Algorithms, chapter 3.6, “The Coupon Collector's Problem”. Cambridge University Press, 1995.

12 PGP homepage. http://www.pgpi.org/.

13 C. G. Plaxton, R. Rajaraman, and A. W. Richa. Accessing Nearby Copies of Replicated Objects in a Distributed Environment. In 9th Annual ACM Symposium on Parallel Algorithms and Architectures (SPAA), 1997.

14 S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A Scalable Content-Addressable Network. In ACM SIGCOMM, 2001.

15 M. Reiter and S. Stubblebine. Authentication metric analysis and design. ACM Transactions on Information and System Security, 2(2), 1999.

16 B. Schneier. Secrets & Lies: Digital Security in a Networked World, chapter 15. Certificates and Credentials. J. Wiley & Sons, 2000.

17 I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan. Chord: A Scalable Peer-To-Peer Lookup Service for Internet Applications. In ACM SIGCOMM, 2001.

Reference to this paper should be made as follows: Karl Aberer, Anwitaman Datta, and Manfred Hauswirth (2004) ‘A decentralized public key infrastructure for customer-to-customer e-commerce’, *International Journal of Business Process Integration and Management*, Vol. X, No. X, pp.000–000.