Using Onion Routing to Secure Peer-to-Peer Supported Business Collaboration

Fabian STÄBER, Udo BARTLANG, Jörg P. MÜLLER
Siemens AG, Corporate Technology, Information and Communications
Otto-Hahn-Ring 6, Munich, D-81739, Germany
Tel: +49 (0)89 636-41619, Fax: +49 (0)89 636-41423
e-mail: {fabian.staeber, udo.bartlang, joerg.p.Mueller}.ext@siemens.com

Abstract: Peer-to-Peer computing provides reliable self-organizing adaptable architectures, fitting perfectly as a foundation for enabling enterprises to seamlessly inter-operate with one another in a loosely coupled fashion. However, the absence of hierarchical structures in P2P architectures raises security challenges, especially in scenarios that are of "co-opetitive" nature, i.e., where the involved business partners may be cooperating and competing, at the same time, and where there is only limited trust between the partners. This paper addresses some of these security issues. We propose an approach using onion routing, which is a well known algorithm, originally used for anonymous email communication. We present a real-world use case regarding business integration in the automotive industry, and show how onion routing can be integrated into our peer-to-peer platform to meet the security requirements of business collaboration environments.

1. Introduction

Over the past few years, the peer-to-peer paradigm received broad attention in academic research and industry. Offering easy to use plug-and-play networks in combination with resilience, reliability, and the capability of decentral management and loosely coupled control, peer-to-peer systems have become target platforms for a growing number of applications, ranging from Internet file sharing over IP telephony to distributed corporate information management. While first progress has been made in studying the applicability, benefits and challenges of the peer-to-peer paradigm in the context of business applications [1], it is still an open topic how business interoperability can benefit from this potential in a *secure* environment.

When implementing business collaboration on top of peer-to-peer systems, security challenges need to be met, as the involved business partners may be cooperating companies and competitors, at the same time. This paper identifies and addresses important challenges to be met when applying peer-to-peer systems in business integration. Starting from a use case in the automotive industry, we describe the security requirements and provide a solution based on the onion routing algorithm.

The use case described in this paper was originally developed in the context of the ATHENA programme [2], an Integrated Project funded by the European Commission, aiming at business interoperability. A key outcome of the analysis of the use case was the need to support a class of application scenarios that we call "coopetitive", i.e., scenarios where business partners cooperate / collaborate in a market in which they are at the same time competitors. In such as scenario, trust between the partners is limited, and there are requirements for security and confidentiality that need to be taken into account during the collaboration process.

We propose a solution enabling confidential communication between business partners in these “coopetitive” scenarios. Our solution builds on and extends *onion routing*, a well-known algorithm, originally implemented in anonymous Remailers [3]. In our case, onion routing is applied to allow business partners to communicate, but to prevent their competitors from learning who is talking to whom.

The paper is structured as followed: In Section 2 we give an overview of the technology used in our framework. Section 3 presents our use case and derives two exemplary threat scenarios. A technical solution is outlined in Section 4; it is compared with related work in Section 5. In Section 6, we present and discuss the results, benefits, and trade-offs from our work; the paper ends with conclusions and outlook in Section 7.

2. Technology Description

The communication platform underlying our work is the Business Resource Management Framework (BRMF) [1], a peer-to-peer framework based on a Distributed Hashtable (DHT) mechanism. From the user’s point of view, the BRMF maintains an abstract collaboration space where business objects (resources) can be published, subscribed, modified, and renewed. To date, the BRMF has been used to manage and share business documents, distributed business events, to facilitate distributed cooperative work on product models, but also to provide a decentral business service registry.

In this Section, we briefly introduce the technical architecture of the BRMF. We will show that the layered architecture makes it possible to replace the underlying peer-to-peer protocol without the need of modifying the onion routing extension or any higher level business application.

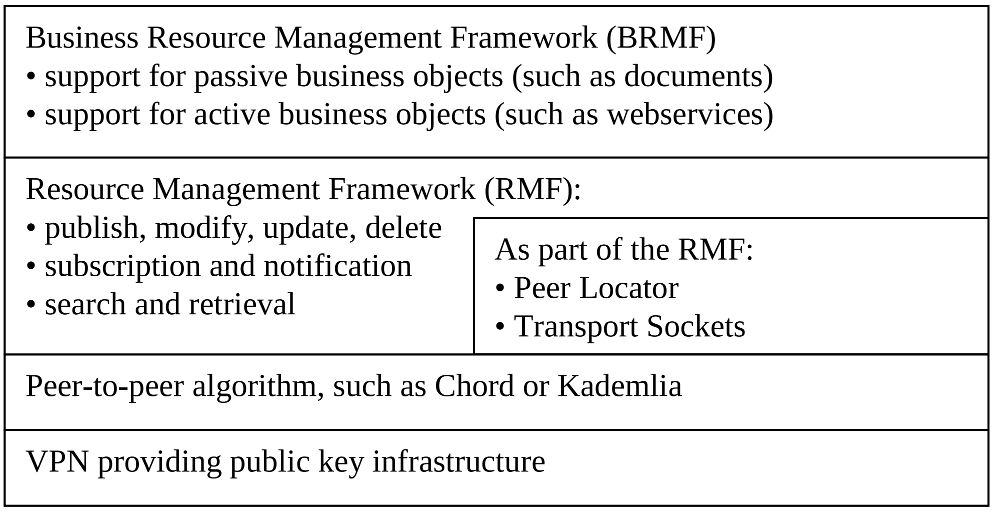
Business Resource Management Framework (BRMF)
• support for passive business objects (such as documents)
• support for active business objects (such as webservices)
Resource Management Framework (RMF):
• publish, modify, update, delete
• subscription and notification
• search and retrieval
As part of the RMF:
• Peer Locator
• Transport Sockets
Peer-to-peer algorithm, such as Chord or Kademlia
VPN providing public key infrastructure

Figure 1: Architecture of a Business Resource Management Framework (BRMF) Environment

The architecture of the BRMF environment is shown in Figure *1*. On the top level the *Business Resource Management Framework* (BRMF) provides explicit support for business objects, maintaining a repository for passive business objects (i.e. documents), as well as active business objects (i.e. web-service lookups).

Underneath the BRMF, the *Resource Management Framework* (RMF) provides an application-independent abstract collaboration space, where resources can be published, searched, and subscribed. It uses a notion of peer locators and transport sockets, abstracting the details of the actual peer-to-peer algorithm in use. The onion routing approach described in this paper is implemented using these features.

Below the RMF, we have the actual *peer-to-peer protocol* implementation, providing the topology and the routing algorithm of the overlay network. Our publish/subscribe algorithm relies on Distributed Hash Tables, as used in Chord [4] or Kademlia [5]. We use Chord in our prototype implementation.

On the lowest level, we propose a *Virtual Private Network* (VPN) to be used as the underlying network infrastructure. This prevents eavesdroppers from intercepting the peer-to-peer protocol data and ensures that no unsolicited participants join the network. The public key infrastructure provided by the VPN for encryption and authentication can be integrated in our security solution.

To sum up, the BRMF provides a layered architectural framework, where onion routing can be implemented independently of the underlying peer-to-peer protocol, and without the need of adapting the higher level business applications.

3. Problem Statement

From the user's point of view, the BRMF environment provides an abstract collaboration space that is implemented on top of an underlying peer-to-peer system. This offers plug-and-play functionality as well as an adaptable resilient architecture. The business partners run equal peers in the BRMF, none of the peers can actually act as a central trusted entity controlling the actions of the others. This absence of hierarchical structures on the peer-to-peer level raises security challenges.

Coming from the overview of a BRMF environment in the previous section, we will now present our use case and analyze the implications at the peer-to-peer layer. We then derive two concrete threat scenarios that we will deal with in this paper.

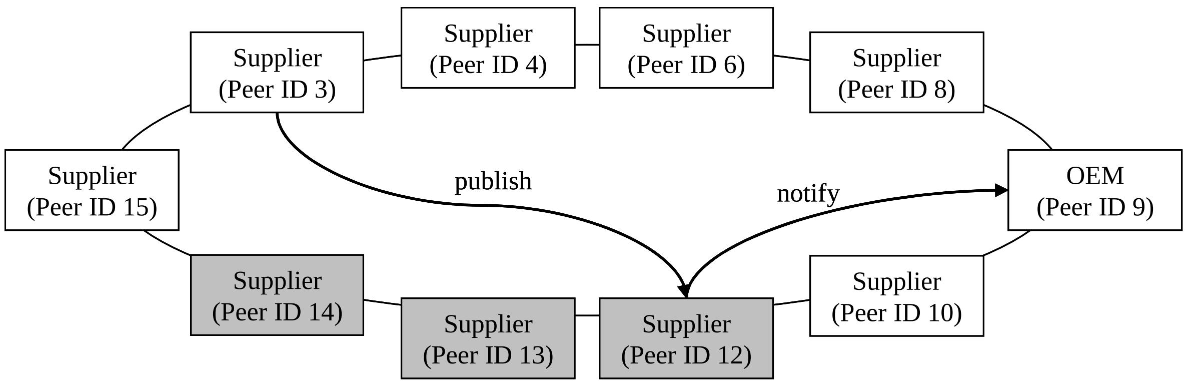

Figure 2: The BRMF Collaboration Ring in the Automotive Example

3.1 Use Case Description

The use case is taken from a car manufacturer's strategic sourcing process. The car manufacturer (OEM, Original Equipment Manufacturer) issues *Requests for Quotation*s (RfQs). After negotiating technical specifications[1] the first tier suppliers send their bids to the OEM.

The business partners interact using the BRMF, which means that the OEM as well as the suppliers provide peers in the peer-to-peer network. The communication is implemented via a publish/subscribe mechanism, enabling resilience if peers go offline.

Unlike flooding-based peer-to-peer systems, the BRMF uses keywords to determine the peer where a certain resource is stored. Each peer is responsible for a certain range of keywords. Additionally, there is a replication group for each keyword, as illustrated in grey. If a peer leaves the network, the next peer in the replication group takes over the corresponding keyword range.

In Figure 2, the peer with id 3 issues a quotation and the peer with id 12 is responsible for the keywords associated with that quotation. All bids for a certain RfQ have the same keyword, such that the OEM subscribing this keyword will receive all bids for this RfQ. As all bids have the same keyword, they are all stored on the same peer, and backup copies are kept in the same replication group.

3.2 Threat Scenarios

When joining the network, a malicious supplier might deliberately choose an id matching the hashed keywords for a certain RfQ[2]. Then all the bids for that RfQ will be published on the supplier's peer. That means that the supplier routes the bids of all of its competitors through its peer. In Figure 2, the attacker would be the peer with the id 12.

That way, malicious competitors can establish intermediate peers, intercepting the quotations sent from suppliers to the OEM. Of course, encryption can be used to make the content of the bids only visible for the OEM. But even though the competitor cannot see the content of the bids, two threat scenarios remain:

1. An attacker can observe whether a certain business rival places a bid, and lower the price in reaction to this knowledge.
2. An attacker can keep track of its competitors and raise the price in case none of them places a bid.

In the following section, we will present a solution that allows us to deal with these scenarios.

4. Our Solution

In the previous section, we derived two concrete threat scenarios when applying BRMF as the underlying collaboration platform in our use case. In this section we show how onion routing can be implemented in a straightforward way to meet these challenges.

The basic assumption underlying our approach is the existence of a public key infrastructure (PKI), such that each peer has a signed key pair. Each supplier must have one and only one signed public key. As each supplier has only one signed public key, it is possible to recognize if a single supplier runs more than one peer. This prevents the so-called Sybil attacks [6]. The PKI can be imported from an underlying VPN, as recommended in Section 2. Alternatively, we can use the PKI from a security extension of the RMF that was originally developed for enabling restricted access to resources.

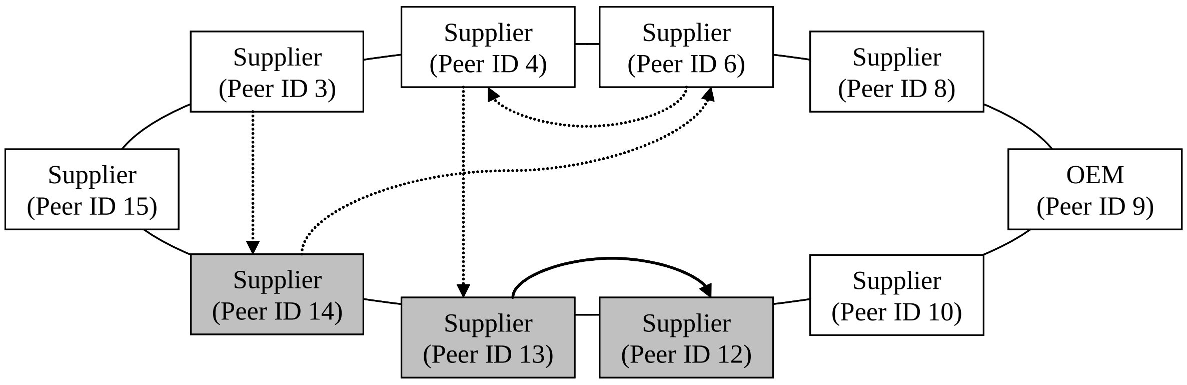

Figure 3: Publishing a Resource using Onion Routing

The basic idea behind onion routing is illustrated in Figure 3. As in the example in the previous section, the peer with id 3 needs to publish a resource on the peer with id 12. Using onion routing, peer 3 does not contact peer 12 directly but chooses a random path through the peer-to-peer network. Then peer 3 encrypts the resource recursively, as shown in Figure 4.

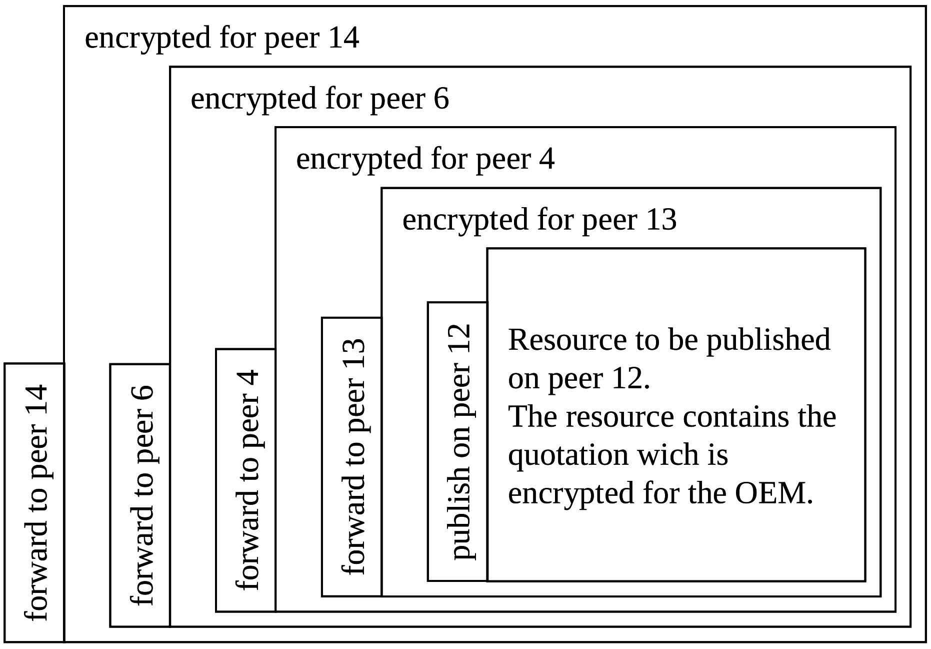

Figure 4: Resource Before Traversing the Onion Path

Each peer on the path decrypts the resource and forwards it to the next hop. The idea is that each node on the path just forwards some opaque, encrypted piece of information from one hop to the next, not being able to tell where the resource originally comes from and where it will finally go to. This is called onion routing, because the decryption is analogous to peeling the layers of skin off an onion. Finally, the last peer on the path publishes the resource. Note that the content of the resource may still be encrypted with the OEM's public key.

The number of hops used for the onion path is a trade-off between performance, reliability, and security. The Mixmaster anonymous Remailer uses a default path length of four hops, which is a reasonable value for our solution as well. In our prototypical implementation for the BRMF, we used a path length of five hops.

The identity of the original sender must not be known to the peers on the path. Therefore, it is important that the sender establishes the onion path silently, without interacting with the peers on the path. Therefore, peers maintain a cache containing peer locators and public keys in order to be able to build the onion path locally.

As a result, onion routing can be used to enable confidential communication between business partners, while the competitors are not able to tell who is talking to whom. The algorithm described here was implemented prototypically in the BRMF supporting our scenario. However, the ideas can easily be applied in other business collaboration platforms as well, supporting "coopetitive" scenarios.

5. Related Work

While security in peer-to-peer networks is a relatively new topic (see [7] for an overview of current security issues in peer-to-peer networks), onion routing itself has existed for about 25 years. In this section, we review other implementations and discuss what features could be implemented in our business collaboration platform, and what features should be left out.

Originally developed for implementing anonymous email communication, onion routing nowadays is used in a large number of applications. Currently, the most popular anonymous Remailer is Type II, as implemented by Lance Cottrell [8]. Type III Remailers [9] are still work in progress. Onion routing is also applied in the Java Anon Proxy [10], which is an http proxy making it possible to surf the Internet anonymously. A more general approach is taken by Tor [11], implementing onion routing on the TCP layer, such that any network application can be anonymized by installing Tor on the underlying TCP layer.

In the peer-to-peer world, the traditional pragmatic focus is on anonymously publishing and downloading information, such as music and video files. Currently, rather sophisticated peer-to-peer systems implementing security features are available, like Gnunet [12], Freenet [13], and Crowds [14]. However, these systems lack the feature of notifications and the flexibility of being customizable to support different types of resources and business applications, which makes them less suited for implementing negotiation protocols than the BRMF.

Looking at the existing implementations, one frequently employed concept is the use of fixed-length messages, dummy traffic and random delays. That way, applications protect themselves against attackers who are able to trace the whole traffic in the entire network. As this is an unrealistic scenario in our use case, we leave this out in favor of simplicity, better performance, and lower overhead.

Another interesting concept is Circuits, as implemented in Tor, which has been mentioned above [11]. When forwarding a message, a random ID is sent together with the message. Each hop on the route maintains a table as shown in Table 1.

incoming id	Sender	Outgoing id	receiver
…	…	…	…

Table 1: Routing Information for Anonymous Acknowledgements Using Circuits

Using that information, reply messages can be routed back along the onion path, while still no hop on the path can tell where the reply message originally comes from and where it will finally be sent to. It would be possible to extend this feature to implement anonymous subscriptions in the BRMF.

Of course, apart from these examples, there are a lot more interesting concepts and related work, and it is worthwhile investigating what features could provide business benefits, and what features should be left out in favour of better performance.

6. Discussion of the Approach

In the previous sections, we presented a use case from the automotive industry regarding collaborative product design, and determined two threat scenarios when using the peer-to-peer based Business Resource Management Framework as the underlying integration platform. We introduced a solution based on onion routing and gave an overview of features of related work that could be included in our implementation.

Devising security architectures, concepts, and solutions always involves trade-offs [15]. In this section we will discuss the key trade-offs entailed by our solution and conclude with the next steps towards implementing a secure peer-to-peer based platform for business collaboration.

6.1 Confidential Communication

By analyzing the use case scenario, we saw that confidential communication among business partners is not guaranteed when using a peer-to-peer system as the underlying collaboration platform. Adding our security extension, we enable confidential communication, making peer-to-peer available for electronic collaboration and marketplace scenarios. By means of this extension, we claim that we have taken one step ahead towards enabling businesses to benefit from decentral management and coordination, self-organization and resilience of peer-to-peer systems without loosing the ability to communicate confidentially.

6.2 Performance

Onion routing relies on routing resources through a random path instead of publishing them directly. We propose using a path length of four to five hops, which means that resources need to be transmitted four to five times, as opposed to a single transmission without onion routing. Additionally, peers need to maintain a cache with current public keys and peer locators, requiring periodical update traffic. Preliminary simulation experiments provide evidence that it is not unrealistic to achieve acceptable system performance using these settings.

6.3 Reliability

When publishing resources directly, the sender gets direct feedback if the resource reached the destination successfully. Using onion routing, the resource is sent to the first hop and there is no way for the sender to learn whether a resource finally reaches its destination successfully, or if a node on the path fails forwarding the resource. Implementing acknowledgements via circuits would solve this problem, as discussed in Section 5. However, doing so would imply other security risks, as malicious peers could acknowledge messages and discard them anyway. Therefore we do not implement acknowledgements in the BRMF, and prefer application level acknowledgements, sent from the OEM directly to the suppliers. Finding an efficient way to solve the problem within a P2P framework is an open research issue.

7. Conclusions and Outlook

There is a lot of potential in deploying peer-to-peer systems in business collaboration. The business benefits of the plug-and-play functionality, adaptability and resilience in combination with the needlessness of central servers are still to be explored.

However, implementing peer-to-peer systems as the underlying collaboration platform in business integration raises security challenges that need to be solved. In this paper, we presented a scenario from the collaborative product design in the automotive industry, applying the peer-to-peer based Business Resource Management Framework as the underlying communication platform. We analyzed the security requirements and presented a generalized solution that can easily be applied for securing any peer-to-peer based auction scenario.

A proof-of-concept implementation shows that our solution greatly enhances the applicability of the BRMF as an adaptive, decentrally manageable, and secure platform for enterprise interoperability.

In this paper, we have analyzed two threat scenarios in a specific use case, and we derived a solution ready to secure collaboration protocols running on top of peer-to-peer based integration architectures suitable for “coopetitive” scenarios such as the collaborative product design scenario described above. Future research will verify and extend this solution by means of a more complete analysis of the security threats[3] in different domains, in order to extend the applicability of peer-to-peer ready to a broader set of scenarios in real-world business integration.

Acknowledgments

Part of the work reported in this paper is funded by the European Commission within the ATHENA IP under the European grant FP6-IST-507849. The authors are solely responsible for the paper's content. The paper does not represent the view of the European Commission nor that of other consortium members.

References

[1] T. Friese, J. Müller, M. Smith, B. Freisleben, "A Robust Business Resource Management Framework Based on a Peer-to-Peer Infrastructure", *Proceedings of the 7th International IEEE Conference on E-Commerce Technology*, IEEE Press, 2005

[2] The ATHENA Integrated Project, http://www.athena-ip.org

[3] D. Chaum. "Untraceable electronic mail, return addresses and digital pseudonyms", *Communications of the ACM,* Volume 24, Issue 2, 1981

[4] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. "Chord: A scalable peer-to-peer lookup service for Internet applications", *Technical Report TR-819, MIT,* March 2001.

[5] P. Maymounkov, D. Mazieres. "Kademlia: A peer-to-peer information system based on the XOR metric", *Proceedings of IPTPS02,* March 2002

[6] J. Douceur, "The Sybil Attack", *1st International Workshop on Peer-to-Peer Systems (IPTPS '02),* Springer, 2002

[7] L. Divac-Krnic, R. Ackerman, "Security-Related Issues in Peer-to-Peer Networks", in *Peer-to-Peer Systems and Applications* by R. Steinmetz and K. Werle (Eds.), Springer, 2005

[8] Mixmaster implementation by Lance Cottrell, http://mixmaster.sourceforge.net

[9] G. Danezis, R. Dingledine, D. Hopwood, N. Mathewson. "Mixminion: Design of a Type III Anonymous Remailer Protocol", *http://mixminion.net,* 2002

[10] JAP: The Java Anon Proxy, http://anon.inf.tu-dresden.de

[11] Tor: An anonymous Internet communication system, http://tor.eff.org

[12] GNUnet: GNU's decentralized anonymous and censorship-resistant P2P framework http://www.gnunet.org

[13] I. Clarke, O. Sandberg, B. Wiley, T. W. Hong, "Freenet, a distributed anonymous information storage and retrieval system", in *ICSI workshop on design issues in anonymity and unobservability*, 2000

[14] M. K. Reiter, A. D. Rubin, "Crowds: anonymity for web transactions", in *ACM Transactions on Information and System Security*, 1998

[15] Bruce Schneier, "Beyond Fear", Copernicus Books, 2003

---

[1] Actually, the negotiation phase is tightly interleaved with an n:m-type cooperative product design process, with multiple negotiations involving multiple RfQs and quotations running in parallel. For the sake of clarity of the use case, we focus on the simple case where a single RfQ is answered with a single Quotation.

[2] In Chord, the Peer ID is not chosen randomly but generated from the IP address, but in that case the attacker can try several IP addresses until he obtains the desired ID. Even if there is a way to prevent this (for example by assigning the id by a central service), the attacker can still get the desired ID by incident.

[3] Some examples are: "How can a malicious node affect the topology of a peer-to-peer network, sending forged routing information to certain other peers?" or "How can a malicious node influence the routing of resources, sending forged lookup responses to certain other peers?"