P2PNS: A Secure Distributed Name Service for P2PSIP

Ingmar Baumgart
Institute of Telematics
Universität Karlsruhe (TH)
Zirkel 2, D–76128 Karlsruhe, Germany
baumgart@tm.uka.de

Abstract

Decentralized Voice over IP networks are a promising alternative to classical server-based SIP networks especially in disaster areas or areas without centralized infrastructure. This paper presents P2PNS, a secure distributed name service for P2PSIP. P2PNS can be used to resolve SIP AoRs to Contact URIs without using DNS or centralized SIP servers. The name service provides several security mechanisms to efficiently prevent identity theft and to ensure the uniqueness of SIP AoRs in a completely decentralized and untrusted network. The proposed two-stage name resolution mechanism allows to efficiently handle frequent IP address changes. Because P2PNS provides a generic name service it is not limited to P2PSIP but can also be used e.g. to build a distributed DNS system.

1. Introduction

An emerging use case for overlay protocols are decentralized VoIP networks. Recently an IETF working group has been formed to develop protocols for the use of the Session Initiation Protocol (SIP) [12] in networks without centralized servers.

Decentralized VoIP networks are especially suitable for use in mobile Ad-hoc networks (MANETs) due to their independence on central servers. They need minimal configuration and can be quickly deployed making P2PSIP and MANETs an ideal combination to establish a communication platform in disaster areas.

In traditional SIP networks the main task of a SIP server is to resolve an *Address of Record (AoR)* to the current IP address (*Contact URI*) of a user. This name resolution usually depends on DNS. In this paper we present a distributed name service using a DHT to resolve AoRs to Contact URIs without relying on DNS and centralized SIP servers. Apart from this decentralized name resolution the call setup is based on the standard SIP protocol. The benefit of this approach is that we can easily connect legacy SIP phones to our P2PSIP network. This connection is accomplished by a SIP proxy located between SIP phone and DHT which handles the name resolution.

Currently there are several other P2PSIP proposals like SOSIMPLE [5], SIPPEER [15], RELOAD [10] and P2PP [1] which are similar to our P2PNS approach. We therefore focus on three aspects in this paper which we think have been neglected by these proposals.

First, we propose several security mechanisms to provide a high level of security in a completely decentralized network without login servers or a PKI. In particular P2PNS provides mechanisms to guarantee the uniqueness of AoRs and to prevent identity theft. These security mechanisms are based on a cryptographically generated *nodeID*, which is used to authenticate overlay nodes.

The second aspect is flexibility: P2PNS is a generic name service not limited to P2SIP, which can be used to resolve arbitrary names to transport addresses. Other applications for P2PNS are e.g. decentralized DNS, decentralized XMPP or decentralized HIP. In P2PNS there is a clear separation between the overlay layer (key-based routing), the data storage layer (distributed hash table), the name resolution layer (P2PNS Cache) and the protocols, that utilize the name service (like SIP or DNS). In this architecture the specification of the key-based routing protocol is independent from P2PSIP and KBR protocol implementations can therefore easily be reused for other peer-to-peer applications.

Finally we propose a two-stage name resolution mechanism similar to a ID/Locator split architecture to efficiently handle frequent IP address changes.

The rest of the paper is organized as follows: In section 2 we provide some background on structured peer-to-peer networks. The requirements for our name service are presented in section 3 followed by the design of our name service described in section 4. Finally we present our ideas for possible next steps in section 6 followed by the conclusion in section 7.

2. Structured peer-to-peer networks

In this section we provide some background on structured peer-to-peer networks. A common service which is provided by all structured peer-to-peer networks is the *key-based routing layer (KBR)* [7]. This layer provides efficient routing to identifiers called *keys* from a large *identifier space*. Every participating node in the overlay chooses a unique *nodeID* from the same id space and maintains a routing table with *nodeIDs* and IP addresses of neighbors in the overlay topology. Every node is responsible for a particular range in the identifier space, usually for all keys close to its *nodeID*. The KBR layer provides *route()* and *lookup()* methods to efficiently route a message to an arbitrary key by successively forwarding the message to overlay neighbors which have a *nodeID* closer to the destination key. In this paper we propose to use the Kademlia [11] protocol as KBR layer, although our findings can also be applied to other KBR protocols.

On top of the KBR we use a *distributed hash table (DHT)*, which is a distributed storage service for storing *(key, value)* data records. The DHT layer provides the two methods *get(key)* and *put(key, value)*. The node responsible for storing a data record with a specific key is discovered by using the *route()* method of the underlying KBR layer.

3. Requirements

The name service P2PNS should fulfill the following requirements:

- The name service should not be limited to P2PSIP, but also support e.g. distributed DNS. Therefore the name service should be independent from the SIP protocol.
- The P2PNS architecture should be completely decentralized. In particular it should not depend on any centralized login servers or other trustworthy authorities.
- The user should be able to choose an arbitrary AoR.
- P2PNS should provide mechanisms to guarantee the uniqueness of AoRs and prevent identity theft.
- P2PNS should support unmodified legacy SIP UAs and provide gateway functionality between P2PSIP and server-based SIP networks.

4. Design

In this section we describe our P2PNS architecture and propose several security extensions for the KBR and DHT layer.

4.1. P2PNS architecture

The P2PNS architecture comprises a name resolution and caching layer (P2PNS Cache) on top of an overlay which provides KBR and DHT services. The KBR service can be provided by any structured peer-to-peer protocol which provides a CommonAPI interface [7] and contains our proposed security extensions. Applications like a SIP proxy connect to P2PNS by using a XML-RPC interface which provides register() and resolve() functions. This modular architecture offers a clean separation of layers and allows to easily exchange the protocols on KBR and DHT layer.

In order to facilitate the use of legacy server-based SIP phones, we decided to employ a proxy architecture. In this architecture every P2PSIP peer consists of a SIP UA, a local SIP proxy as well a P2PNS implementation. The proxy is used as a location server for resolving AoRs to Contact URIs by using the P2PNS services.

To facilitate the interconnection of P2PSIP and server-based SIP networks we propose to use AoRs of the form *username@p2pname.org*. The *username* part can be freely chosen by the user whereas the domain part *p2pname.org* is fixed and used to identify the P2PSIP network. In order to connect the P2PSIP network to the server-based SIP network the domain *p2pname.org* should contain a SRV DNS record pointing to several of the P2PSIP proxies which forward SIP INVITEs to the appropriate P2PSIP nodes. In pure P2PSIP networks DNS is not used at all.

4.2. Two-stage name resolution

P2PNS uses a two-stage approach to resolve a AoR to the current Contact URI. For this purpose every peer chooses once a 160 bit *nodeID* for joining the overlay. This *nodeID* is retained even if the peer changes its IP address or leaves the overlay from time to time. The KBR layer allows us to efficiently resolve the *nodeID* to the current IP address of a peer. By choosing an AoR of the form *nodeID@p2pname.org* the P2PSIP proxy could use the KBR service to forward an SIP INVITE to the destination proxy without using a DHT.

Because using the *nodeID* as AoR is against our requirement of letting the user choose an arbitrary name as AoRs we additionally store a mapping from the arbitrary AoR to the corresponding *nodeID* in the DHT. In this case the name resolution layer first queries the DHT for the *nodeID* of the destination node and in a second step resolves this *nodeID* to the node's current IP address.

Instead of using this two-stage name resolution approach, it is also possible to directly store an AoR to IP address mapping in the DHT. But due to the security mechanisms proposed in section 4.4 storing and modifying data

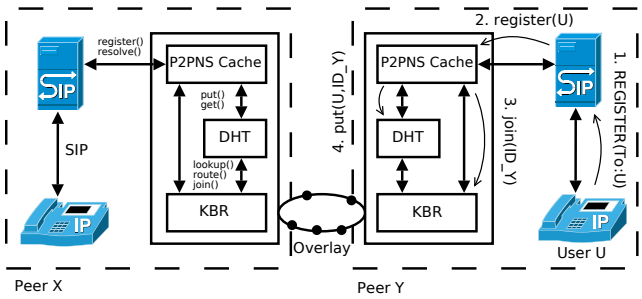

Figure 1. Registration of an AoR with P2PNS

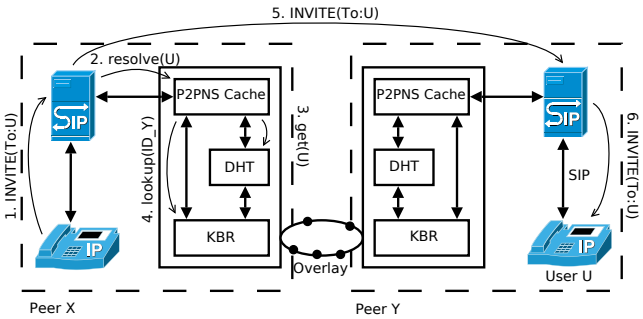

Figure 2. Resolving an AoR with P2PNS

records in the DHT is very bandwidth consuming, because data records are replicated on several nodes. However by using the two-stage approach it is not necessary to modify data records, because the AoR to *nodeID* mapping doesn't change. So an IP address change has an effect on the KBR layer only, which can be handled very efficiently.

To reduce communication overhead and lookup latency when resolving the same AoR several times, the static AoR to *nodeID* mappings are cached locally. The *nodeID* to Contact URI mapping may also be cached. Because this mapping can be outdated, it has to be verified. This is done by trying to directly contact the destination node using the stored destination IP address. If this verification fails, the mapping is refreshed by using the KBR lookup() method.

Figure 1 illustrates an example of a AoR registration. In the first step the UA at peer Y sends a SIP REGISTER message with the AoR *U* to the local P2PSIP proxy. The proxy then connects to the P2PNS XML-RPC interface and calls register(U). If the P2PNS layer is not already connected to the overlay, it joins with *nodeID ID_Y*. Finally the P2PNS layer stores the AoR to *nodeID* mapping in the DHT.

Figure 2 shows how the user at peer X establishes a call to the AoR *U*. At first the UA sends an SIP INVITE to the local P2PSIP proxy. Subsequently the proxy queries P2PNS by a resolve(U) call. The P2PNS layer first fetches the corresponding *nodeID* for the AoR *U* from the DHT (if the mapping is not already cached). In the next step the obtained *nodeID* gets resolved to the current IP address of peer Y. Finally the SIP INVITE message gets forwarded to the UA via the proxy at peer Y.

4.3. KBR security

The security of the P2PNS architecture largely depends on the security of the KBR layer. As shown in [6] KBR protocols have to fulfill three requirements to provide a high level of security. On the basis of these requirements we decided to use Kademlia as KBR protocol and extended it by several security enhancements [4]. In the following these requirements and security enhancements are summarized:

4.3.1. Secure nodeID assignment

Most important it should be hard for an attacker to generate a large number of *nodeIDs* (Sybil attack [8]) or to choose a particular *nodeID* freely (Eclipse attack [14]). In P2PNS every node has to generate a public key pair and calculates its *nodeID* by applying a cryptographic hash function $H(x)$ on its public key k_{pub}. To impede the generation of a large number of *nodeID* we additionally make use of crypto puzzles. A simple crypto puzzle is given below:

1. Generate a new public key pair (k_{priv}, k_{pub}).
2. Calculate $H(H(k_{pub}))$ and check, if the first c bit are 0.
3. If the condition in step 2 is not true, repeat step 1. Otherwise the crypto puzzle is solved and the *nodeID* is $H(K_{pub})$.

This approach has several benefits compared to the usual approach to generate the *nodeID* by applying a hash function on the IP address of the node. First the node may keep the *nodeID* if the IP address changes. Furthermore the public key approach can be used in networks with NAT (Network Address Translation), in which several nodes share the same public IP address.

The public key pair (k_{priv}, k_{pub}) is used in the following to authenticate overlay signaling. For this purpose overlay messages are signed with k_{priv}. In this way the receiving node may use the public key k_{pub} attached to the message to verify the authorship.

Using crypto puzzles as defense against Sybil attacks is currently the most promising approach in completely decentralized networks. However crypto puzzles only make a Sybil attack harder and cannot completely prevent it. If the attacker has sufficient computing ressources to solve a large number of crypto puzzles particularly small networks as well as the bootstrap phase are still vulnerable to attacks.

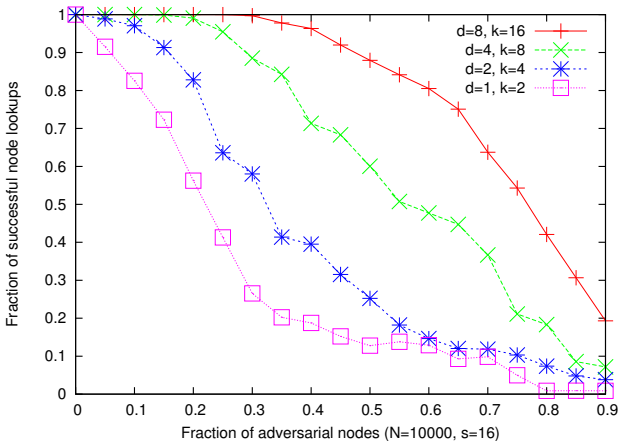

Figure 3. Fraction of successful node lookups when using d disjoint paths in a Kademlia network with malicious nodes

In these cases the *nodeID* assignment should additionally be restricted by using certificates of an offline CA.

4.3.2. Lookup over disjoint paths

As second requirement the overlay should provide several disjoint and preferably short paths to all destination keys to successfully deliver messages in presence of malicious nodes. The number of disjoint paths depends particularly on the employed overlay topology (e.g. ring, hypercube or de Bruijn graph). The Kademlia protocol is based on a hypercube topology and provides the bucket size parameter k, which can be used to tune routing table redundancy to the required level of security.

We studied the influence of disjoint paths on lookup success in a network with malicious nodes by using the OverSim framework [3]. The simulation results for Kademlia are illustrated in Figure 3. These results show the significant increase in lookup success by using d disjoint paths. The bucket size parameter k was chosen in such a way that the overlay topology provides sufficient redundancy for d disjoint paths.

Most overlay protocols can be used with recursive as well as iterative routing. In P2PNS iterative routing is used to ensure the resulting paths are really disjoint. Furthermore with iterative routing the originator of the lookup can constantly monitor the lookup progress as shown in Figure 4. Yet iterative routing exhibits the disadvantage of roughly doubling the time for a lookup to finish in comparison to recursive routing.

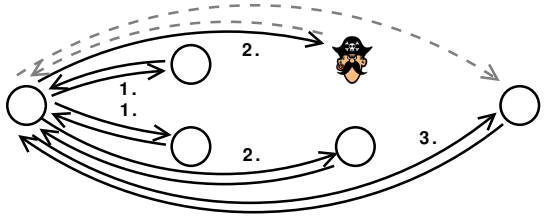

Figure 4. Successful iterative lookup in spite of malicious nodes when using disjoint paths

4.3.3. Secure routing table maintenance

An important security property of KBR protocols is the robustness of the signaling protocol for routing table maintenance in the presence of malicious nodes. As long as the *nodeID* selection is limited, Kademlia is very robust against adversarial routing table manipulations due to is implicit stabilization by incoming lookup requests. Because Kademlia uses a least-recently-used replacement strategy for routing table updates, new nodes are only added if older nodes fail. Therefore Kademlia is not vulnerable to the flooding of bogus routing updates once the network is bootstrapped.

4.4. DHT security

The proposed security mechanisms in section 4.3 are the basis for providing a secure DHT service. Yet the DHT layer has to fulfill additional requirements to secure the stored AoR to *nodeID* mappings:

- Data records may only be deleted or modified by the owner of the record.
- Data records should be replicated on several nodes to inhibit manipulation by single malicious nodes.
- The DHT should be secure against insertion DoS attacks.

In order to prevent the unauthorized modification of data records the DHT layer additionally stores the *nodeID* of the owner along with the data. If a node wants to subsequently modify a data record, it has to sign the modification request with its private key k_{priv}. The receiver of the request has verify the signature and to ensure that $H(k_{pub})$ coincides with the *nodeID* of the data record's owner.

The node, that is responsible for storing a data record is determined by means of the key of the record. In this case the key is the hash value of the AoR. In order to prevent users from choosing an already existing AoR, the DHT only

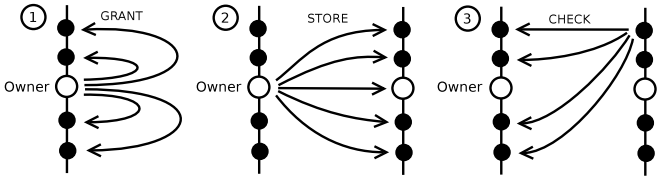

Figure 5. Secure data storage in a DHT limiting the number of allowed data records per owner

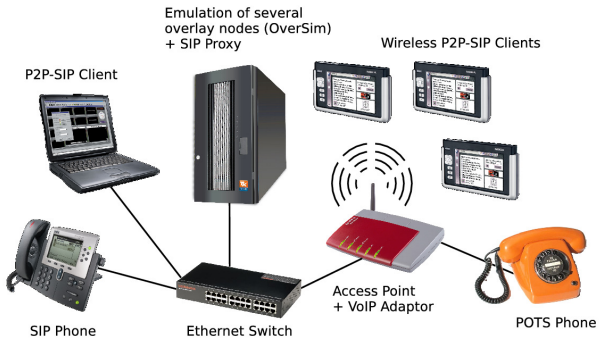

Figure 6. Architecture of the P2PSIP demonstrator

stores a single data record for a each key. Consequently the user how stores his AoR first is eligible for this name.

Data records are replicated on several nodes, because a malicious node may arbitrarily tamper with locally stored data records. The replicas are stored on neighbor nodes close to the key as these nodes can be efficiently discovered by a single KBR lookup.

A peer resolves an AoR to *nodeID* mapping by querying all replicas in parallel. Thereupon the peer makes a majority decision on all received replies to determine the most plausible destination *nodeID*. In order to handle churn every newly joined node first requests all data records in his responsibility from his neighbor nodes and stores them locally.

Finally the DHT has to be protected against adversarial flooding of insertion requests. This is important because the verification of the signature of a STORE message is computational expensive. Moreover the storage of unnecessary data records consums valuable peer ressources. To compensate for the computational ressources for verifying the signature of a STORE message, the requesting node has to solve the following crypto puzzle:

For the key k of the data record determine an appropriate b, so that the first c bits of $H(k \oplus b)$ are equal to the first c bits of the own *nodeID*. The constant c is used to specify the complexity of the puzzle and b is the solution of the puzzle.

The crypto puzzle makes the insertion of a large number of data records harder, but doesn't completely prevent an insertion DoS attack. Therefore we additionally limit the allow number of data records per owner by using the approach illustrated in Figure 5. To store a new data record the owner O sends a GRANT message to all neighbors close to his own *nodeID* after solving the crypto puzzle. These neighbors store all keys of the data records that O has already stored in the DHT. If the maximum number of allowed data records per owner is exceeded the GRANT message is rejected. In a second step the node O sends a STORE message with his data record containing the AoR to *nodeID* mapping to all replicating nodes. These nodes use a CHECK message to verify if the neighbor nodes of O have authorized the storage and finally store the data record locally.

The proposed security mechanisms make the storage and modification of data records rather expensive in terms of computational and communication costs. But by using the two-stage approach of section 4 the static AoR to *nodeID* mapping has to be stored only once when new AoR is registered for the first time. If a node later change its IP address or temporarily leaves the network, this is efficiently handled by the KBR layer without involving complex DHT operations. In [13] the author proposes to directly generate AoRs by applying a hash function to a public key. A major drawback of this approach is the AoR can't be chosen freely by the user and using a large random number as AoR is hard to remember.

5. Implementation

We implemented the proposed architecture as prototype in a demonstrator [2]. The demonstrator consists of several Nokia 800 internet tables with 802.11g interfaces, a Linux laptop, an unmodified legacy SIP phone as well as an old POTS phone. Figure 6 illustrates the architecture of the demonstrator.

Each of the Nokia 800s and the laptop is running an unmodified SIP UA, a P2PSIP proxy based on *openser* and an OverSim instance. OverSim [3] is an overlay network simulation framework and features the reuse of protocol implementation in real networks. We use OverSim in our demonstrator to provide the P2PNS name service to the P2PSIP proxy by using an XML-RPC interface. The OverSim instances communicate using the 802.11g interfaces and build a logical Kademlia overlay topology.

The legacy SIP phone and the old POTS phone are connected to the P2PSIP network by an external P2PSIP proxy running on a dedicated node. In order to increase the number of overlay nodes there is an additional OverSim process which can be used to emulate additional overlay nodes

making the overlay network size more realistic. Finally this OverSim process features a GUI to illustrate the emulated network traffic.

6. Future work

At the moment we are working on a detailed performance evaluation of the proposed P2PNS architecture in our OverSim framework. On focus of this evaluation is to study the latency involved in establishing a secure P2PSIP call using P2PNS compared to using a server-based SIP network.

The proposed security extensions for Kademlia could also be added to other KBR protocols. Therefore we want to compare the performance and security properties of Kademlia to the performance of the Broose [9] protocol, which is a KBR protocol similar to Kademlia but based on a de Bruijn graph.

7. Conclusion

In this paper we presented P2PNS, a distributed name service for decentralized Voice over IP networks. In contrast to previously proposed architectures for P2PSIP the P2PNS name service provides a high level of security without relying on any centralized login servers or a PKI.

The proposed proxy architecture allows a seamless integration of legacy SIP UAs and avoids modifications to the complex SIP protocol stack. The security mechanisms of P2PNS efficiently ensure unique AoRs and prevent identify theft in a completely decentralized and untrusted network. The modular design of P2PNS allows to easily exchange the protocols on KBR or DHT layer and eases the integration in existing networks. Because P2PNS provides a generic name service it is not limited to P2PSIP but can also be applied e.g. to build a distributed DNS system.

Acknowledgment

This research was supported by the German Federal Ministry of Education and Research as part of the *ScaleNet* project 01BU567. The author likes to thank Sebastian Mies for his valuable contributions to this work.

References

[1] S. Baset, H. Schulzrinne, and M. Matuszewski. Peer-to-peer protocol. work in progress, draft-baset-p2psip-p2pp-01, November 2007.

[2] I. Baumgart, B. Heep, and S. Krause. A P2PSIP Demonstrator Powered by OverSim. In *Proceedings of 7th IEEE International Conference on Peer-to-Peer Computing (P2P2007), Galway, Ireland*, pages 243–244, Sept. 2007.

[3] I. Baumgart, B. Heep, and S. Krause. OverSim: A flexible overlay network simulation framework. In *Proceedings of 10th IEEE Global Internet Symposium (GI '07) in conjunction with IEEE INFOCOM 2007, Anchorage, AK, USA*, pages 79–84, May 2007.

[4] I. Baumgart and S. Mies. S/Kademlia: A Practicable Approach Towards Secure Key-Based Routing. In *Proceedings of the International Workshop on Peer-to-Peer Networked Virtual Environments 2007 (P2P-NVE 2007) in conjunction with ICPADS 2007, Hsinchu, Taiwan*, Dec. 2007.

[5] D. A. Bryan, B. Lowekamp, and C. Jennings. Sosimple: A serverless, standards-based, p2p sip communication system. In *First International Workshop on Advanced Architectures and Algorithms for Internet Delivery and Applications (AAA-IDEA 2005), 15 June 2005, Orlando, Florida, USA*, pages 42–49, 2005.

[6] M. Castro, P. Druschel, A. Ganesh, A. Rowstron, and D. S. Wallach. Secure routing for structured peer-to-peer overlay networks. *SIGOPS Oper. Syst. Rev.*, 36(SI):299–314, 2002.

[7] F. Dabek, B. Zhao, P. Druschel, J. Kubiatowicz, and I. Stoica. Towards a common api for structured peer-to-peer overlays. In *Proceedings of the 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03)*, volume Volume 2735/2003, pages 33–44, 2003.

[8] J. R. Douceur. The sybil attack. In *IPTPS '02: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 251–260, London, UK, 2002. Springer-Verlag.

[9] A.-T. Gai and L. Viennot. Broose: a practical distributed hashtable based on the de-bruijn topology. In *Fourth International Conference on Peer-to-Peer Computing, 2004*, pages 167–174, aug 2004.

[10] C. Jennings, B. Lowekamp, E. Rescorla, and J. Rosenberg. Resource location and discovery (reload). work in progress, draft-bryan-p2psip-reload-02, November 2007.

[11] P. Maymounkov and D. Mazires. Kademlia: A peer-to-peer information system based on the xor metric. In *Peer-to-Peer Systems: First International Workshop, IPTPS 2002 Cambridge, MA, USA, March 7-8, 2002. Revised Papers*, volume Volume 2429/2002, pages 53–65, 2002.

[12] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. Sip: Session initiation protocol. RFC3261, June 2002.

[13] J. Seedorf. Using cryptographically generated sip-uris to protect the integrity of content in p2p-sip. In *Third Annual VoIP Security Workshop, June 2006, Berlin, Germany*, 2006.

[14] A. Singh, T.-W. J. Ngan, P. Druschel, and D. Wallach. Eclipse attacks on overlay networks: Threats and defenses. In *In Proceedings of INFOCOM 06, Barcelona, Spain. April 2006*, 2006.

[15] K. Singh and H. Schulzrinne. Peer-to-peer internet telephony using sip. In *NOSSDAV '05: Proceedings of the international workshop on Network and operating systems support for digital audio and video*, pages 63–68, New York, NY, USA, 2005. ACM Press.