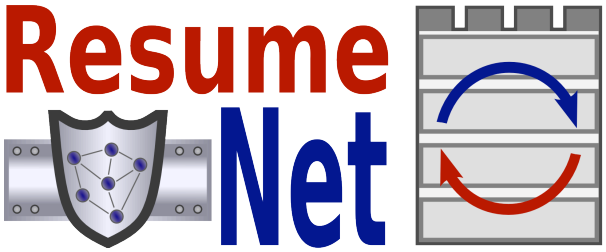
Resilience and Survivability for future networking: framework, mechanisms, and experimental evaluation



Deliverable number	3.3
Deliverable name	P2P Overlays and Virtualization for Service Resilience
WP number	3
Delivery date	08.31.2010
Date of Preparation	6/9/2010
Editor	Ali Fessi (TUM)
Contributor(s)	Andreas Fischer (UP), Yahya Al-Hazmi (UP)
Internal reviewer	Nathan Evans (TUM)

Summary

P2P networking and virtualisation are promising techniques among the resilience mechanisms which have been investigated in ResumeNet for providing resilient network services. A major and common motivation behind both these techniques is that they provide an abstraction from the underlying hardware resources. Thus, they allow for isolating failures at lower layers or at least reducing the Mean-Time-To-Repair (MTTR) if failures occur. Therefore, P2P networking and virtualisation address inherently several aspects of the ResumeNet D^2R^2-DR strategy, particularly the inner control loop. This deliverable describes the results of our investigations in these two topics. We take Voice over IP (IP) as an example of a critical application. Based on reliability theory and traces from the Skype network, we provide a quantification of the reliability that can be provided by a P2P network for VoIP session setup.

In contrast to a pure P2P network approach, we propose a supervised P2P network approach to address the security issues in P2P networks. The "supervisor" provides the peers with verifiable identities. It is involved in the session setup under normal operation. However, even if the supervisor is unavailable, the service can still be provided by the P2P network. A supervised P2P network can be considered as a solution between server-based and pure P2P-based signaling solutions. The goal is the combination of the advantages of both architectures leading to improved reliability and security. An extensive security threat analysis evaluates to what extent the supervisor addresses the security issues of P2P networks and which additional security mechanisms are still required.

Virtualisation allows for migration of services with a variety of migration strategies. However, resilience requirements of services can vary widely. Thus, there is a need for managing service migration in a resilient way, picking the best strategy for service migration. In the paper attached to this deliverable, we present an architecture for resilient service migration, taking into account changing service requirements and different properties of migration strategies.

Contents

1 Introduction

P2P networking and virtualisation are among the important mechanisms which have been investigated for providing network resilience at the service level. The overall ResumeNet architecture for resilient services has been described in the ResumeNet deliverable D3.1b. The results of the research work on virtualisation are summarized in the paper attached in the appendix of this deliverable. The paper is intended for publication at the *International Symposium on Integrated Network Management 2011 (IM'2011)*. Thus, the main part of this document focuses on P2P networks.

P2P networks are decentralized by design. They are built to cope with churn, i.e., with peers joining and leaving the network regularly. Also, data stored in a P2P network is *replicated* at multiple nodes in the network. Nodes can detect that their neighboring peers have left the network and thus, can *autonomously recover from stale routing table entries*. Moreover, P2P networks offer *geographic diversity*. Inherent decentralization, data replication, autonomous recovery from stale routing table entries, and geographic diversity, these are properties which make P2P networks attractive for building resilient network services.

Nevertheless, it is important to estimate the reliability that can be provided by a P2P network, notably in case the P2P network should be used for critical applications such as IP telephony. For example, a important question is whether the P2P network is able to provide "5 nines" reliability, which means a reliability of 99.999%. Moreover, a second question is whether an attacker would be able to invalidate the reliability assumptions. For example, storing an object at k peers provides redundancy, but is useless if all k peers are controlled by the same single malicious node. A central authority is required in the network to provide the same security guarantees as in server-based services, and to appropriately benefit from the reliability advantages of P2P networks without scarifying security. However, an extensive evaluation is still required to analyse to what extent the central authority can help to address these issues.

The rest of this document is structured as follows. Section 2 describes an implementation of the concept of *supervised P2P networks* in the context of SIP and VoIP. A quantification of the reliability of the system is provided based on reliability theory and traces from the Skype network. Section 3 discusses the security challenges faced when using a P2P network for critical applications and how far a supervised approach can help to address these issues. Further mechanisms to make the system resilient against attacks are provided and analysed. Section 4 provides related work and Section 5 concludes this document.

2 Coooperative SIP (CoSIP)

Telephony is a critical service which needs to be protected in everyday life as well as particularly in case of large scale disasters. The transition from the PSTN/ISDN to VoIP and the use of the Internet as a common medium for telephony and data raises new resilience requirements on the Internet. Security threats that show up due to the use of the same medium for the audio/video data and the legacy Internet applications, e.g, DoS attacks, spam over IP telephony (SPIT), are among the major problems.

Apart from security, reliability is another major issue which deserves more attention with the wider deployment of VoIP. The complexity of the infrastructure for VoIP is one of the main reasons for this lack of reliability. The infrastructure includes SIP registrars, SIP proxies, AAA servers, DNS servers, DHCP servers, routers, firewalls and other network components, which require complex configuration. Even careful design leads frequently to inter-dependencies between these components. Network and service failures may propagate quickly. Moreover, reliability is affected by Denial-of-Service (DoS) attacks and flash crowds.

A number of incidents of VoIP service interruption have been reported in the press, e.g.,

- Subscribers of the German VoIP provider "1&1" experienced service failure on January 5^{th} 2010 for up to four hours [Hei10c]. It is assumed that the root cause is a DoS attack on the DNS servers [Hei10a].
- Another service failure was experimented by "1&1" subscribers on November 11^{th} 2009. According to the provider, the failure was due to network congestion and had to be remediated by manual traffic shaping [Hei10b].
- In August 2007, the P2P-based VoIP service Skype was unavailable for two days. Most of the functionality of Skype can occur in a P2P manner. However, the login process requires the Skype servers, which were unavailable. The businesses of many enterprises that used Skype on a daily basis were affected [New07].
- A disruption in the infrastructure of the VoIP service at "United Internet" - a consortium of 4 large VoIP providers in Germany - occurred in July 11^{th} 2006. It was not possible to make phone calls for two hours [Hei06].

We mention also some other prominent examples of large scale failures of telephony services. These are not VoIP-based, but the root causes are software failures, e.g.,

- On April 21^{st}, 2009, a failure during a Software update of the Home Location Register (HLR) of T-Mobile rendered the T-Mobile network in Germany down for 4 hours.Around 40 million subscribers were affected. During that time it was not possible to make phone calls, to send text messages or to use the Internet connections[1].
- The failure of a server at Deutsche Telekom on Monday October 29^{th} 2007 [wel07] made phone calls to and from other providers unfeasible. Some calls were even routed incorrectly, i.e., phone calls were established to the wrong Callee.

[1] unless people kept their mobile phone switched on and did not leave the same cell (3G or GSM cell). This is because in all other cases, re-authentication of the user based on the (U)SIM card is required, which involves the HLR, which was down

These prominent examples indicate the clear need for better resilience, notably better reliability and security for telephony as a service. In this section, we present Cooperative SIP (CoSIP) a supervised P2P network approach which addresses the two critical resilience aspects of VoIP signaling mentioned above, namely reliability and security. CoSIP is a hybrid architecture based on a P2P network cooperating with central servers. The P2P network consists of SIP endpoints that organize themselves in a DHT. Both the DHT and the server manage user registration and session establishment in parallel.

While the DHT provides additional service reliability and robustness against DoS attacks, the server provides improved security and performance for the overall architecture. Our new architecture uses both technologies in parallel to combine advantages from both concepts, leading to improved reliability, security and performance.

2.1 Background

2.1.1 Session Initiation Protocol (SIP)

SIP [RSC+02] is a protocol standardized by the IETF for setting up multimedia sessions, notably VoIP sessions. It can also be used for Instant Messaging (IM) [Ros04]. In the last few years SIP has dominated over the ITU H.323 protocol. It has been integrated into the 3GPP IP Multimedia Subsystem (IMS) [3GP08].

The most relevant components of a SIP network are User Agents (UA), proxies and registrars. A SIP UA may either generate requests or process requests from other UAs. In the former case, the UA is called User Agent Client (UAC). In the latter case, the UA is called a User Agent Server (UAS). A SIP registrar is a server that processes the registration of a UA to a certain location. The SIP registrar may use a location database and a AAA server to manage registrations and location information of its UAs. SIP proxies process and forward requests of SIP UAs, working together with SIP registrars in order to establish sessions between two UAs. Due to their close functionality, SIP registrars and proxies are often co-located together.

SIP uses of the Session Description Protocol (SDP) [HJP06] to carry the session parameters, e.g., codecs, IP and port where to send the media streams. A SIP message consists of SIP headers and eventually a SDP body. Not all SIP messages carry a SDP body. For example, an `INVITE` request and the corresponding `200 OK` response carry the parameters for the session in the SDP body. A SIP `REGISTER` request and the corresponding `200 OK` response do not.

When the signaling for establishing the session is completed, UAs can start to send media data to each other. Media data can be exchanged directly between two UAs, typically using the Real-Time Protocol (RTP) [SCFJ03]. In some cases, a relay node is used, e.g., due to NAT traversal problems, or if all the media data is routed via a single node for access control. More details on SIP can be found, e.g., in [Cam01, Joh03],

2.1.2 P2P-based SIP (P2PSIP)

The basic idea of P2PSIP is as follows: Instead of using SIP servers, proxies and registrars which manage the location of users and moderate session establishment requests, P2PSIP uses a P2P network in which each node acts as a registrar.

For the illustration of the functionality of P2PSIP, we assume the usage of a DHT as a P2P network (in contrast to an unstructured overlay as the Kazaa network [LRW03] and the Skype network [BS06]). The node identifier (node ID) in the DHT may be the hash

value of its IP address, possibly combined with a port number. Another possibility is to use the hash value of the public key. When Alice registers with her current location, the data (`alice_IP:alice_port`) is stored in the DHT under the key `H(alice@example.org)`, where `H` is a system-wide hash function. Then, Alice locates the nodes in the DHT who are responsible for the key
`H(alice@example.org)`. Generally, nodes responsible for a key in a DHT are the nodes whose identifiers are the closest to the key according to the distance metric used in the DHT[2]. These nodes are called *replica nodes*. Alice asks the replica nodes to store her contact data. When Bob needs to establish a session with Alice, he sends a lookup message with the key `H(alice@example.org)`. The lookup message is propagated in the DHT until it reaches at least one of the replica nodes storing Alice's contact data. One or more replica nodes respond to the request. Then, Bob can initiate the SIP signaling directly with Alice without involving any servers.

Storage is based on a soft-state, i.e., Alice needs to refresh her contact data in the DHT periodically, e.g., once an hour. Otherwise, the replica nodes will consider the data as outdated and will delete it. In case Alice goes offline and some of the replica nodes are still storing Alice's contact data, Bob will be able to find Alice'contact data which were last stored in the DHT, but will not be able to reach Alice.

In March 2007, the IETF working group P2PSIP was chartered. The focus of the working group is to push SIP processing towards the endpoints and to keep the required infrastructure minimal, cf. [P2P]:
The Peer-to-Peer (P2P) Session Initiation Protocol working group (P2PSIP WG) is chartered to develop protocols and mechanisms for the use of the Session Initiation Protocol (SIP) in settings where the service of establishing and managing sessions is principally handled by a collection of intelligent endpoints, rather than centralized servers as in SIP as currently deployed.

David Bryan[3] published a document shortly afterwards with an overview of the motivations behind P2PSIP [Bry07]. One of the main motivations is to decrease OPEX of VoIP providers. An other motivation is the easing of ad hoc communication. According to Schulzrinne [Sch08], reliability is also a motivation given that P2PSIP should not depend on central components or DNS for session establishment.

2.2 CoSIP Overview

In this section, we present the CoSIP concept as well as example application scenarios.

2.2.1 Concept

Unlike the P2PSIP approach discussed in the IETF, we follow a supervised P2P network approach, combining the advantages of P2P and SIP client/server architectures in order to provide better reliability, security and performance. The SIP server cooperates with the SIP UAs to manage user registrations and session establishments. The current locations of UAs are stored at the server and are additionally propagated in the P2P network (Figure 1). Lookup requests are likewise resolved by the SIP server and by the P2P network concurrently (Figure 2).

[2]Different DHT algorithms use different distance metrics, e.g., Euclidean distance, Hamming distance, longest prefix match, etc.

[3]The P2PSIP WG co-chair at the time of writing this deliverable.

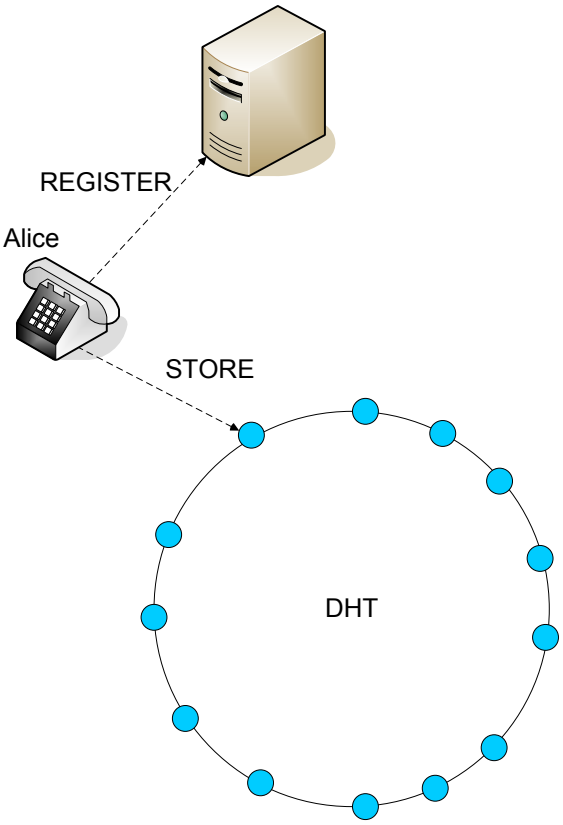


Figure 1: Registration of a SIP UA with CoSIP.

As a choice for the P2P network, we use a DHT in order to achieve efficient routing and avoid flooding requests. Legacy SIP UAs that do not support CoSIP can be connected to the DHT via an adapter that we call "CoSIP proxy" (see Section 2.3). The same holds for weak devices with less CPU or bandwidth. They are represented in the P2P network by a CoSIP proxy running on a more powerful machine. In both cases, a CoSIP proxy processes user registrations as well as session establishments for the UAs behind it.

Figure 1 and Figure 2 show the basic functionality of CoSIP. When Alice's UA registers to the SIP registrar, it also store her contact data in the DHT:

```
STORE(H(alice@example.org), alice_IP:alice_port)
```

This data will be propagated in the DHT and can be used afterwards to resolve the SIP URI of Alice to her current location. By the time another peer, let's say Bob, needs to initiate a session with Alice (Figure 2), Bob sends an `INVITE` message towards the SIP server. Additionally, Bob computes the hash value of Alice's URI and locate Alice's contact data in the DHT:

```
(alice_IP:alice_port) = GET(H(alice@example.org))
```

The `GET` request is propagated in the DHT until one of the replica nodes storing the data with the key `alice@example.org` responds.

A response from the DHT may take longer depending on the routing algorithm used for the DHT, the number of peers and the stability of the DHT. However, in case the server is

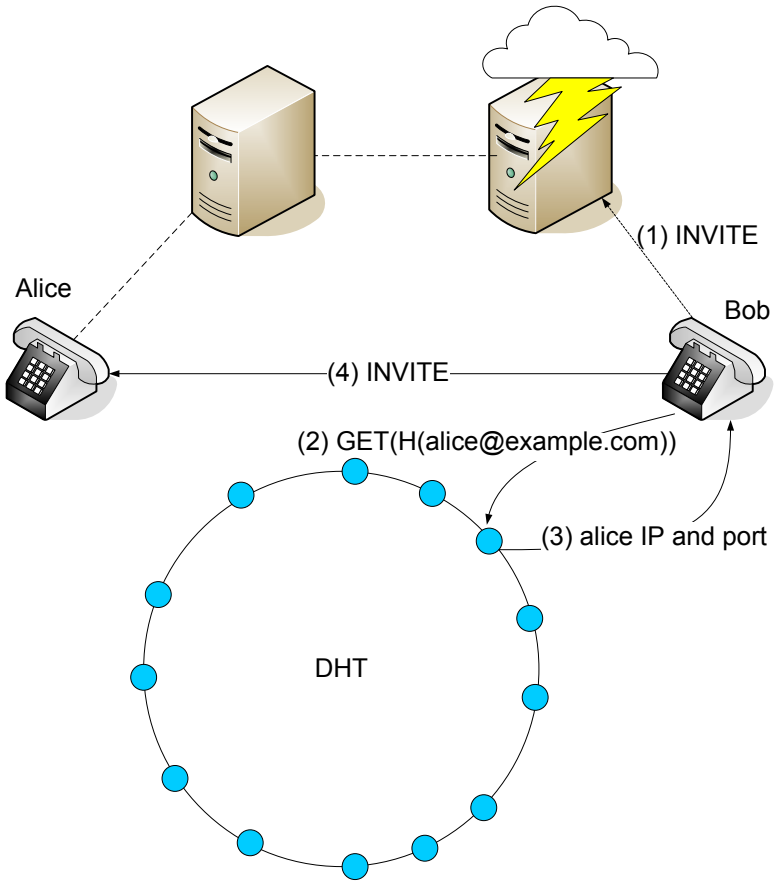


Figure 2: CoSIP session establishment in case of a server failure.

unreachable due to a network or service failure, or the server is undergoing an attack or an overload situation, the response from the DHT can be very useful. When Bob receives the required information to contact Alice from the DHT, he sends an `INVITE` message directly towards Alice and the session establishment can be completed. In other words, in case the server is overloaded or unreachable, the DHT serves as backup. This provides a significant improvement to the reliability of the SIP service compared to traditional SIP. On the other side, CoSIP provides improved security compared to P2PSIP.

Concluding this section, centralized SIP infrastructures have a lack of reliability and are vulnerable to DoS attacks; P2PSIP networks are hard to secure and are vulnerable to a number of attacks such as Sybil attacks, eclipse attacks, partition attacks, or SPIT. Therefore, CoSIP is intended to fill the gap between these two solutions by combining them in order to benefit from their respective advantages.

2.2.2 Application Scenarios

An application scenario of CoSIP is sketched in Figure 3. A large-scale VoIP network, e.g., an enterprise or university SIP network. SIP UAs communicate with each other during normal operation and set up a DHT. Server downtimes due to failures or maintenance can be bridged by CoSIP.

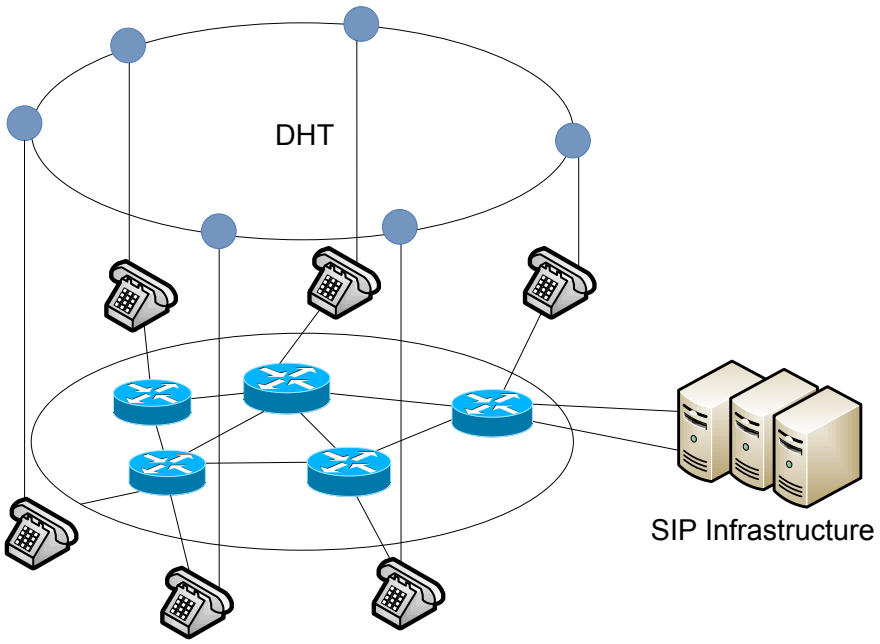


Figure 3: CoSIP operation in an enterprise network.

Figure 4 shows another application scenario of CoSIP. Small Office and Home Networks (SOHO) are connected to an Internet and VoIP provider via DSL routers. Each DSL router contains a CoSIP proxy which acts as an outbound proxy for end devices in its SOHO and implements the additional CoSIP functionality. The CoSIP proxies communicate with each other and organize themselves in a DHT. In the regular case, the SIP infrastructure of the VoIP provider is used to establish sessions between end devices in different SOHOs. In case the SIP infrastructure is temporarily unavailable, the DHT acts as backup and end devices can still establish phone calls.

In both scenarios described above, CoSIP provides a low-cost solution for significantly improving the reliability of the VoIP service. It is a proactive solution which does not require any challenge detection mechanisms or manual configuration when a failure at the infrastructure occurs.

2.3 Prototype Implementation

In this section, we provide an overview of our proof-of-concept implementation of CoSIP.

2.3.1 Design Decisions

On the main decisions is to leverage existing software components where possible and be compliant with standard SIP clients.

Support of Different SIP Clients Software We implemented CoSIP as a local SIP proxy that processes the SIP signaling of one or more SIP UAs. Implementations of the SIP UA do not need to be aware of CoSIP. The SIP UA just needs to be configured with the CoSIP proxy as an outbound proxy.

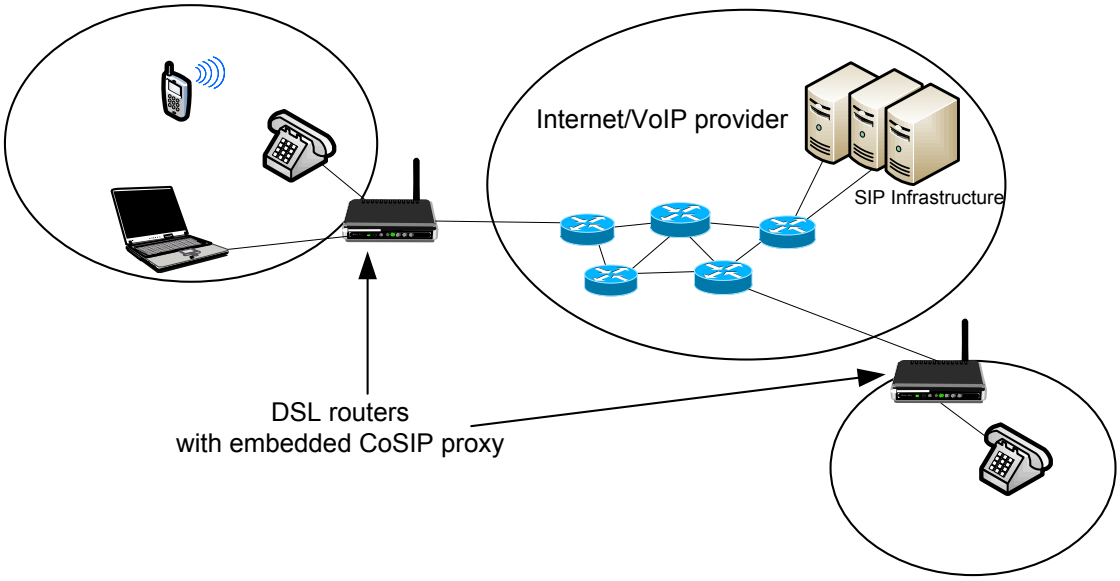


Figure 4: CoSIP operation in an Internet/VoIP provider network.

Choice of a DHT Our CoSIP implementation supports pluggable DHT implementations.

Bamboo: Our CoSIP implementation supports the *Bamboo* DHT [RGRK04]. Bamboo uses the concept of the Pastry DHT [RD01a] with improved routing and maintenance algorithms in order to cope with high churn rates. Bamboo needs less traffic for the overlay maintenance. For the communication between the CoSIP proxy and the DHT node, we use XML-RPC for performing `STORE` and `GET` requests. The XML-RPC interface is provided by Bamboo to simplify the integration of Bamboo into other projects.

Kademlia: Additionally, CoSIP is interoperable with the *Kademlia* DHT [MM02]. We integrated our CoSIP implementation with the Kademlia implementation *entangled*[4].

One of the main motivations for using Kademlia is that it has been the only DHT algorithm which is actually used in practice. More precisely, the KAD P2P network [SENB07] is based on Kademlia. The Kademlia protocol is based on four RPCs only `FIND_NODE`, `FIND_VALUE`, `STORE` and `PING`.

A Supernode Approach It is well known that due to the different capacities of peers in a P2P network, such as CPU, memory and connectivity, there should be a differentiation between powerful peers, which are called the Supernodes, which are connected to the DHT, and weak peers that are connected indirectly to the DHT via the Supernodes. This contributes to the stabilization of the DHT. In CoSIP, we support the Supernode approach by having several SIP UAs connecting to a CoSIP proxy. The CoSIP proxy deals with a few SIP UAs and represents them in the P2P network. However, SIP UAs may also connect to several CoSIP proxies at a time in order to avoid that the CoSIP proxy becomes a single point of failure by itself.

Putting Everything Together Based on the design decisions presented above, the architecture results into different software components communicating together as presented in

[4] `http://entangled.sourceforge.net/`

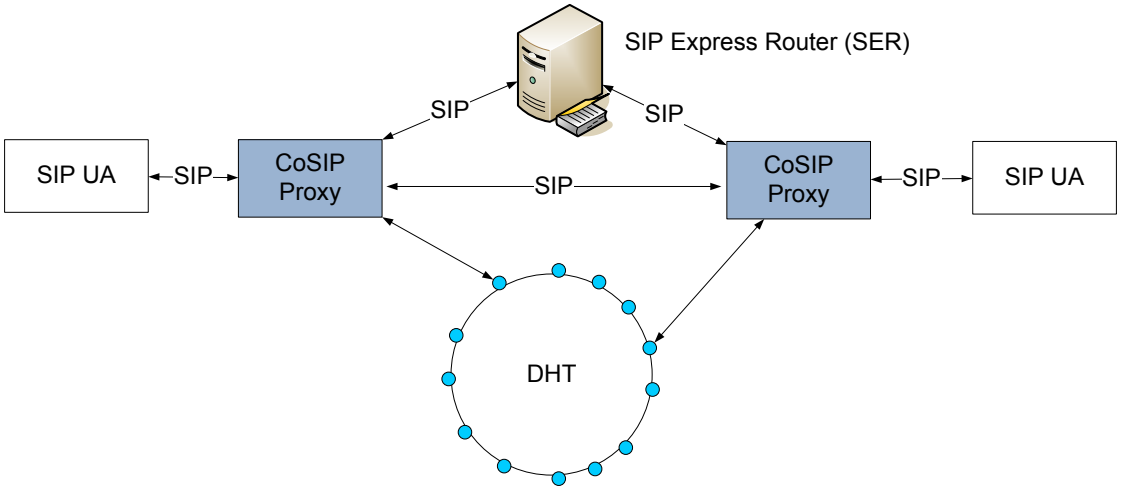


Figure 5: Architecture of a SIP network with a CoSIP proxy.

Figure 5. The SIP UAs use standard SIP to communicate with the SIP proxies. They are unaware of the use of CoSIP or the DHT. The CoSIP proxies use SIP to communicate with the SIP server and among each other. As a SIP server, we use the SIP Express Router (SER) [SIP]. The CoSIP proxy was implemented in Python.

We successfully tested our CoSIP proxy implementation with SER as a server, and Kphone [Sou], Ekigacite [Incb] on Linux as well as XLite [Inca] and QuteCom (former WengoPhone) [Qut] on Windows.

Support of P2P-based SIP As a side effect of our implementation of CoSIP, it is easy to configure the functionality of CoSIP to support a P2P mode without a SIP server. This purely P2P mode can be used, e.g. in small networks where the social contact between the users may make a central authority unnecessary and, e.g., self signed certificates may be used. We also performed some tests with CoSIP in a P2P mode with OpenDHT [RGK+05] and it worked fine. Therefore, our implementation of CoSIP supports three different operation modes:

- DHT-only mode: here the CoSIP proxy makes `STORE`s and `GET`s out of `REGISTER` and `INVITE` methods coming from UAs. No SIP server is involved.
- Cooperative mode: this mode is the main idea behind CoSIP where both the DHT and the SIP server are involved.
- Server-only mode: here the CoSIP proxy forwards the SIP signaling between UAs and the SIP server. Except adding and removing a `VIA` header, no modification to the SIP messages or further processing is undertaken.

Figure 6 shows a screenshot of our CoSIP proxy implementation. The proxy is running in DHT-only mode. It has received a SIP `REGISTER` message from the SIP UA (Ekiga) and performed a successful registration in the DHT.

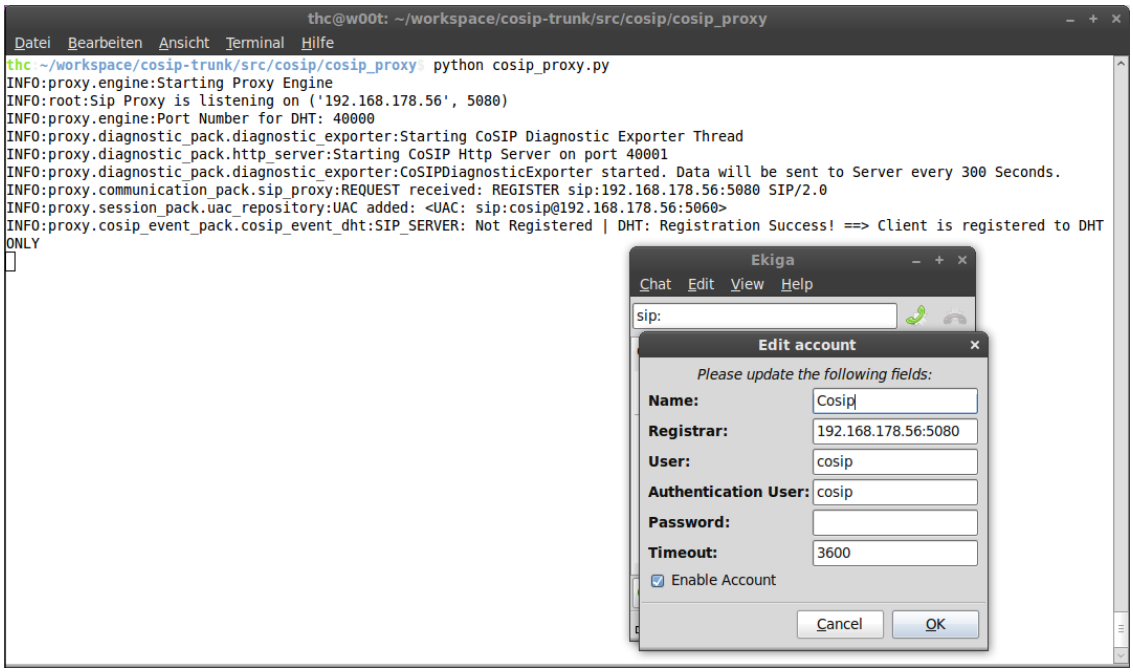


Figure 6: Screenshot with a SIP UA (Ekiga) and a CoSIP proxy running in P2P mode in the background.

2.3.2 Message Processing

If a CoSIP Proxy receives a `REGISTER` from one of its UAs, it adds a VIA header to the message and forwards the modified message to the SIP Server. Furthermore, it stores the contact data of the user in the DHT.

```
STORE(H(alice@example.org), alice_IP:alice_port)
```

Two different timers *i)* for the server registration and *ii)* for the DHT `STORE` RPC are started. If the server does not respond in time, we assume that the server is unreachable for some reasons[5], and register the UA to the DHT only. Registration in the DHT fails in the unlikely case that the CoSIP proxy looses connectivity to the rest of the DHT. In case of success of either registration at the server or at the DHT, the CoSIP proxy responds with a `200 OK` SIP message to the UAC.

As we will see in Section 3, the CoSIP proxy may need to wait for a successful registration at the SIP server before the registration in the DHT is possible. This is the case if the CoSIP proxy needs to acquire a certificate for the UA upon successful registration. Using this certificate, the contact data stored in the DHT can be integrity-protected. However, in case the certificate is still valid, registration in the DHT can be initiated simultaneously with the registration at the server.

Session Establishment with CoSIP If the CoSIP proxy receives an `INVITE` message from a UAC, it adds a `VIA` header to the message and forwards it to the SIP server. It sends a `100 Trying` message to the UAC back. Furthermore, the CoSIP proxy tries to resolve the SIP

[5]Message loss is processed separately at the SIP transaction layer.

URI of the Callee, let's say bob@example.org, to the location of the Callee's CoSIP proxy using the DHT:

```
(bob_IP:bob_port) = GET(H(bob@example.org))
```

Two different timers are initiated to limit the response time of the SIP server and the DHT.

- If the server responds first, the response is forwarded to the UA. A subsequent response from the DHT is suppressed.
- If the DHT responds first, a subsequent response from the server is suppressed. The data received from the DHT is used to send the INVITE message directly to the Callee's CoSIP Proxy. The Callee's CoSIP Proxy forwards the INVITE message to the Callee. The response from the Callee (e.g., 200 OK) traverses both CoSIP proxies back to the UAC.
- If neither server nor DHT respond in time, the CoSIP proxy sends an error message 408 Request Timeout to the UAC.

2.3.3 High-Level State Machines

A CoSIP proxy keeps state information for each UA to process user registration and session establishment. Note however that the CoSIP proxy is not a full-fledged stateful SIP proxy according to RFC 3261 [RSC+02]. It rather keeps a minimal state to coordinate the parallel communication with the server and the DHT.

The state machine of the CoSIP proxy is organized in a modular and hierarchical approach. The CoSIP proxy can deal with one or more UACs behind it by logically separating their state machines. Furthermore, for each UA the state machine is separated according to the SIP sessions. We differentiate between REGISTER and INVITE session state machines. As in standard SIP, different sessions can be differentiated according to the Call-ID.

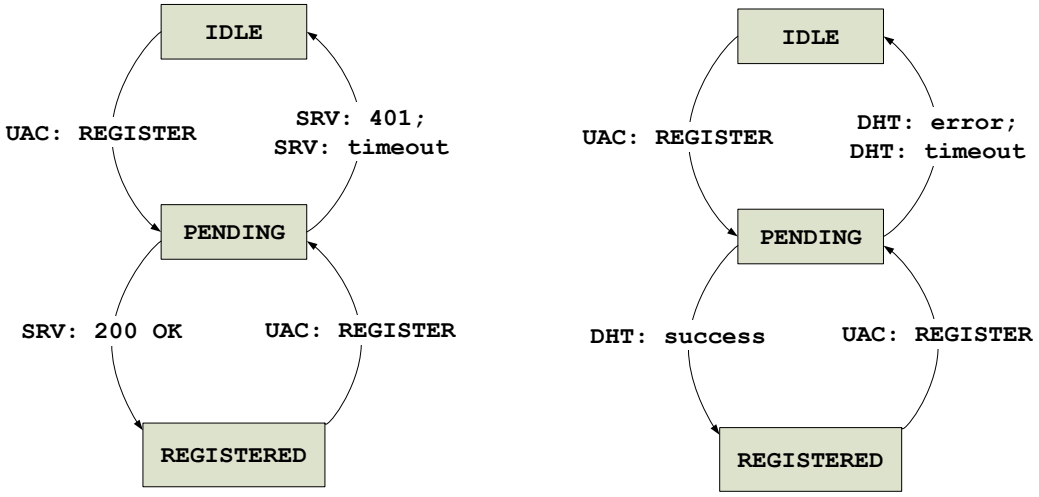
Moreover, each session state machine is separated according to the current state of the communication with *i)* the server and *ii)* the DHT. For example, the REGISTER session state machine shown in Figure 7 is separated into two sub-state machines SRV.* and DHT.*. The state of the session is the compound state of both sub-state machines. For example, if the state is (SRV.REGISTERED $\wedge$ DHT.REGISTERED), this means that the registration has successfully been performed at both the server and the DHT. This results into a clean and unambiguous hierarchy:

UAC $\rightarrow$ session (REGISTER or INVITE) $\rightarrow$ sub-state machine (SRV or DHT)

For each event, notably an incoming message or a timeout, the CoSIP proxy identifies the sub-state machine uniquely and performs the required processing.

REGISTER Session State Machine A REGISTER session is created, if a UA sends a REGISTER message via the CoSIP Proxy to the SIP server. The initial state of the REGISTER session is (SRV.IDLE $\wedge$ DHT.IDLE). The CoSIP proxy forwards the REGISTER message to the SIP server, starts a SRV Timer for limiting the server response time and switches the state of the SRV sub-state machine to SRV.PENDING.

If the server requires authentication, it responds with a 401 Unauthorized message. In this case, the state is set back to SRV.IDLE. By the time the next REGISTER message is received



(a) SRV.* sub-state machine (b) DHT.* sub-state machine

Figure 7: CoSIP REGISTER State Machine.

(which should contain the appropriate authentication credentials), the state is switched back to SRV.PENDING. If a 200 OK message is received from the server in state SRV.PENDING, the state is switched to SRV.REGISTERED. If the SRV Timer expires in state SRV.PENDING, we assume that the server is currently not reachable. The registration at the server fails.

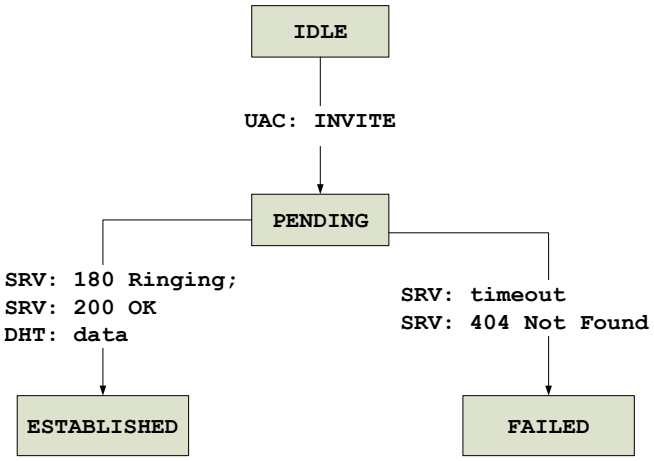
As for the DHT sub-state machine, the CoSIP proxy performs a STORE request and moves to the state DHT.PENDING. If the DHT registration succeeds, the state is set to DHT.REGISTERED. In the unlikely case that registration at the DHT fails, the state is reset to DHT.IDLE.

The registration of a UA to a SIP server expires within a certain period. Thus, the UA needs to renew the registration periodically. If the CoSIP proxy receives a REGISTER message, a new registration cycle is started and the state of both sub-state machines is rest to PENDING.

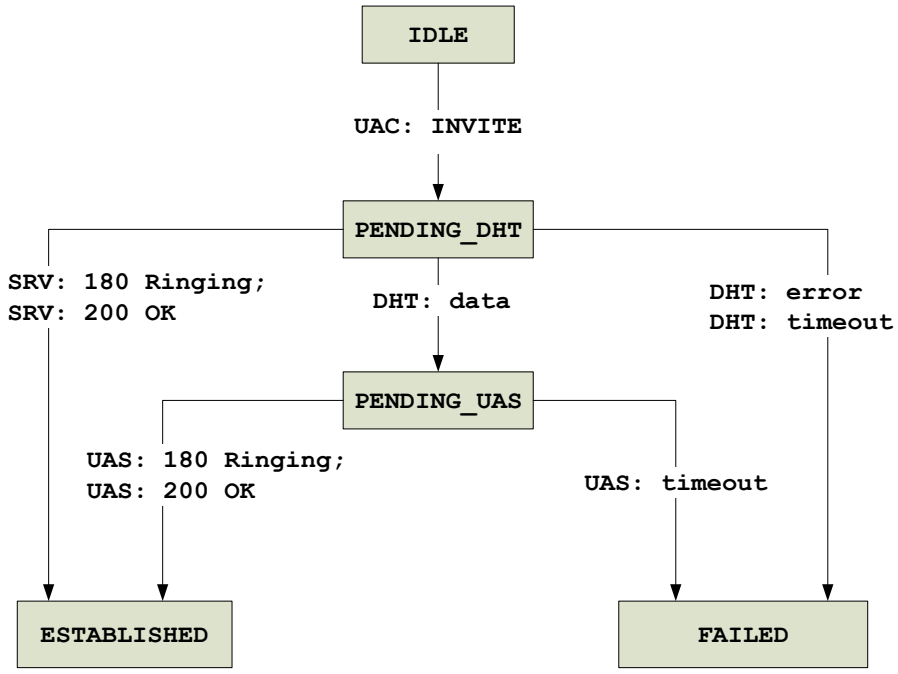
INVITE **Session State Machine** Upon receipt of an INVITE message from a UAC, the CoSIP proxy creates a new INVITE session with two sub-state machines SRV and DHT. Both sub-state machines start in the respective state IDLE. The CoSIP proxy switches the SRV sub-state to PENDING immediately after forwarding the INVITE message to the SIP server. A SRV Timer is started. The SRV sub-state is switched to ESTABLISHED upon receipt of a 180 RINGING or 200 OK response from the server. and to FAILED upon a SRV Timeout event.

As for the DHT sub-state machine, the CoSIP proxy initiates the lookup in the DHT upon the initialization of the INVITE session and switches the DHT sub-state immediately to PENDING_DHT. If the lookup in the DHT fails, e.g., either because the Caller's CoSIP proxy is encountering connectivity problems to the DHT, or no data in the DHT is found, then the DHT sub-state is switched to FAILED. If the DHT lookup is successful, the CoSIP proxy sends the INVITE message to the Callee's CoSIP proxy and switches the DHT sub-state machine to PENDING_UAS.

In the state PENDING_UAS, upon receipt of a 180 RINGING or 200 OK response from the Callee's CoSIP proxy the sub-state is switched to ESTABLISHED. Upon a UAS Timeout event, the sub-state is switched to FAILED. The case of UAS Timeout occurs, e.g., if the Callee has gone offline and its contact data stored in the DHT is outdated.



(a) SRV.* sub-state machine



(b) DHT.* sub-state machine

Figure 8: CoSIP INVITE State Machine.

Finally, in both sub-state machines, the state is moved to ESTABLISHED as soon as either the server or the DHT responds. This guarantees that if a response is received from the server, then any subsequent response from the DHT is suppressed and if a response is received from the DHT is received, then any subsequent response from the server is suppressed. The motivation behind this is to avoid race conditions in the signaling at the Caller UA and make the "forked" signaling at the CoSIP proxy fully transparent to the UA.

2.4 Reliability Analysis

One of the main goals of CoSIP is to improve the reliability of the SIP signaling compared to server-based SIP signaling. In this section, we provide a quantitative analysis of the reliability of CoSIP based on reliability theory.

2.4.1 Reliability Theory

Reliability theory provides tools for estimating the reliability of a whole system by estimating the reliability of the single units/components of the system. Let T be the *time to failure* of a unit, i.e., the time elapsed between when the unit is put into operation until it fails for the first time. T can be assumed to be continuously distributed with a density function $f(t)$ and distribution function:

$$F(t) = Pr(T \leq t) = \int_0^t f(u)\, du \tag{1}$$

The reliability $R(t)$ is the probability that the unit will be still operating at time t:

$$R(t) = 1 - F(t) = Pr(T \geq t) \tag{2}$$

A structure of units is *series* if the operation of the structure depends on the operation of all units in this structure. A *parallel* structure is a structure which operation requires at least one of the units operating.

Let a structure consisting of k units with independent failures[6] and equal reliabilities $R_i(t) = R(t)$ for all units $i = 1, \ldots, k$. If the structure is series, the reliability of the structure is

$$R_\wedge(t) = R_1(t)R_2(t)\ldots R_k(t) = R^k(t) \tag{3}$$

If the structure is parallel, the reliability of the structure is

$$\begin{aligned} R_\vee(t) &= 1 - (1 - R_1(t))(1 - R_2(t))\ldots(1 - R_k(t)) \\ &= 1 - (1 - R(t))^k \end{aligned} \tag{4}$$

2.4.2 Modeling CoSIP with Reliability Theory

We model a CoSIP network as a system which consists of multiple units, which are the peers plus the server. The time to failure T of a peer is the time interval between the point of time when the peer goes online until the point of time when it leaves the network. In other words, T is the peer lifetime.

Modeling the reliability of CoSIP is a challenging task for different reasons:

- Modeling the reliability of the server: we are actually considering a complex system which consists eventually of multiple SIP servers, registrars, proxies, DNS servers, AAA servers, firewalls, etc. as a single unit. Modeling the reliability of such a complex system is not possible if the reliability of the single units within that system and the exact dependencies between them are unknown. Therefore, we will restrict our analysis to the case where the server is unreachable by the SIP UAs for any reason, and compute the reliability of CoSIP in case of server failure.

[6] which is a dominant assumption in reliability theory

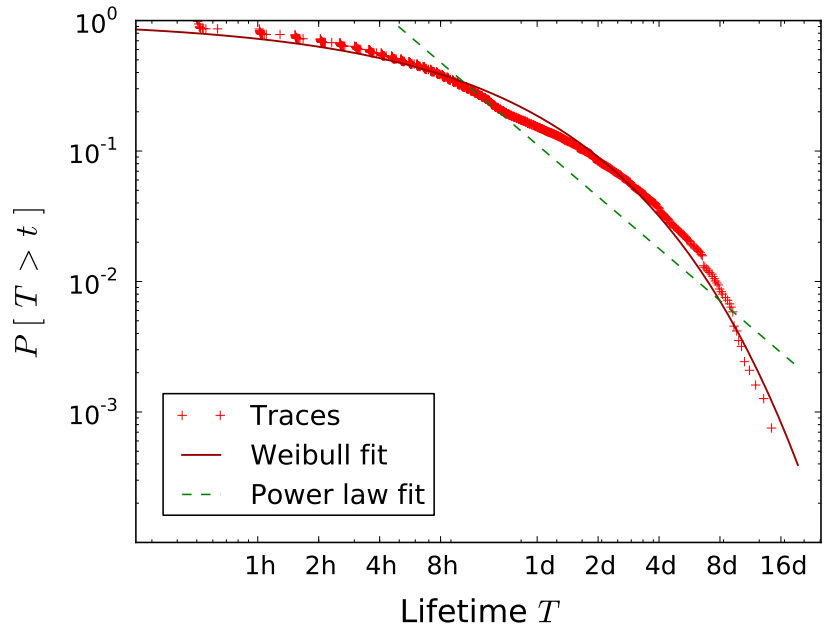


Figure 9: Complementary CDF of the supernode lifetime distribution in Skype.

- Modeling the reliability of the peers: There has been different studies of the peer lifetime in P2P networks, notably KAD [SENB07] and Skype [GDJ06]. However, it is not straightforward that any of these studies is useful to model the peer lifetime in a CoSIP network. KAD is a file sharing network and not VoIP. Skype is a VoIP application. But it is running mainly on PCs/laptops. Thus, Skype shows a high number of online peers during working days and middays, while peers in a CoSIP network could be running, e.g., on some fixed hardphones which are permanently online, or on mobile smart phones, which may change their IP addresses frequently. Nevertheless, Skype is the most similar application to CoSIP where measurement data are available. The Skype traces published in [GDJ06] will help us to estimate the reliability of CoSIP. The reliability analysis of the KAD network in [SENB07] will be also helpful for a comparison.

A Reliability Model based on Skype Traces: Guha et al. [GDJ06] collected traces by monitoring 4000 nodes participating in the Skype supernode network for one month beginning September 12, 2005. The traces are available under `http://www.cs.uiuc.edu/homes/pbg/availability/`. The authors in [GDJ06] plot the complementary CDF (CCDF) of the lifetime of the Skype supernodes in order to detect a power-law relationship. We use the same traces to detect whether the supernode lifetime can be modeled with a Weibull distribution instead. Figure 9 shows the CCDF of the supernode lifetime together with a Weibull fit and and a power-law fit.

It is clear from Figure 9 that the Weibull fit is better suited than a power-law fit. The scale λ and the shape α parameters of the Weibull distribution are approximately equal to 8.84 and 0.52 respectively.

From the definition of reliability (See equation 2), it follows straightforward that the CCDF

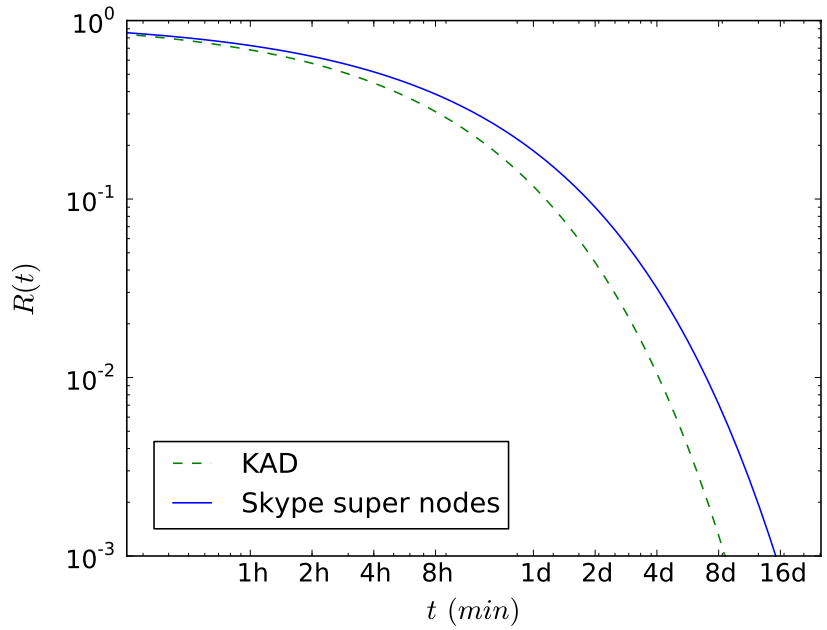


Figure 10: Comparison of the reliability of Skype supernodes and KAD peers.

of the supernode lifetime is its reliability:

$$P[T > t] = R(t) \tag{5}$$

Thus, the reliability of a Skype supernode can be modeled based on the Weibull distribution

$$R_{Skype\_supernode}(t) = e^{-(\frac{t}{\lambda})^{\alpha}}; \quad \alpha = 0.52 \quad \lambda = 8.84 \tag{6}$$

The median peer life time is about 5.1 hours. Note that the median in this case is greater than the median value reported for the KAD network [SENB07] which is about 3 hours ($181min$). The reason is that the Skype traces of Guha et al. include only the Skype supernodes which are more stable. According to [SENB07], the peer lifetime in KAD follows a Weibull distribution

$$R_{KAD\_peer}(t) = e^{-(\frac{t}{\lambda})^{\alpha}}; \quad \alpha = 0.545 \quad \lambda = 5.95 \tag{7}$$

Figure 10 shows the difference between the estimated reliability of KAD peers according to the model derived in [SENB07] and the estimated reliability of Skype supernodes according to the Weibull fit that we derived above.

Now back to the CoSIP reliability model. Let $R(t)$ be the reliability of an arbitrary peer at a point of time t, i.e., the probability that an arbitrary peer is online until t. A UA Bob refreshes his contact data in the DHT periodically (with a `STORE` RPC) with a refreshing period e.g., $\tau = 1h$, in order to make sure he remains reachable in the CoSIP network with high probability. If Bob sends a `STORE` RPC to a replica node Carol at $t = k\tau, k \in \mathbb{N}$, and receives an acknowledgment message from Carol, Bob can deduce that Carol is online[7] Thus,

[7]Under the assumption that the acknowledgment message is integrity-protected.

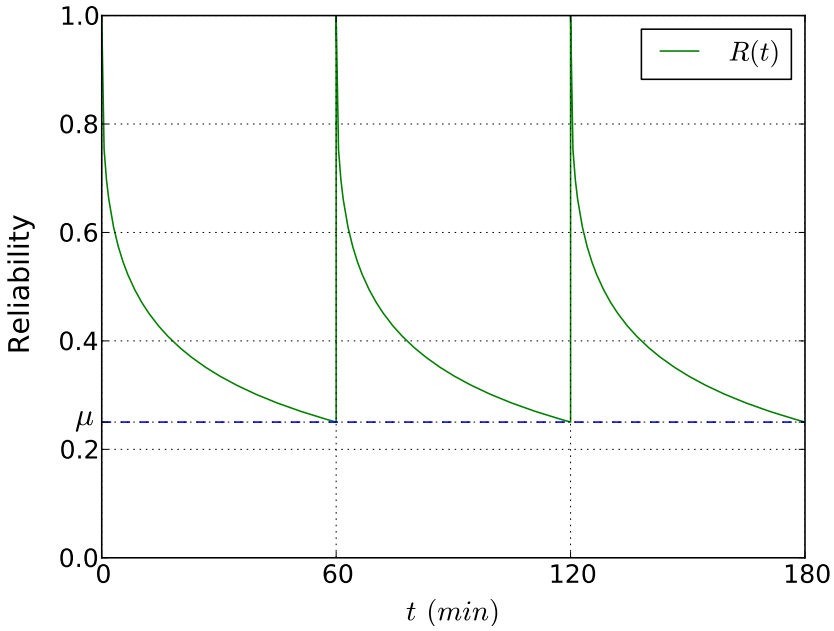


Figure 11: Example reliability of a single peer p_i with periodic refresh. $\tau = 1h$.

the probability that the peer Carol is online at that point of time $t = k\tau$ is equal to one. Then, this probability decreases over the time to the minimum value at the end of the refreshing period. An example of this behavior with a refreshing period $\tau = 1h$ is shown in Figure 11. Let μ be the minimum reliability of an arbitrary peer at the end of each refreshing period τ:

$$\mu = \liminf_{t \to (k+1)\tau} R(t) \quad k \in \mathbb{N} \tag{8}$$

μ can be estimated autonomously by Bob through measurements. It is the probability that if another UA (peer) is observed online at a point of time t, the UA will remain online until $(t + \tau)$. The value μ can be considered as a metric for the churn in the network. If the measured μ is too low, then the UA may have to decrease the refreshing period τ, and thus increasing μ.

Furthermore, the following assumptions are required for our reliability analysis:

- We assume that all peers are cooperative, i.e., as long as a peer is online, it will perform storage requests from other peers.
- We assume that peers/UAs leave and join the network independently. A UA which leaves the network deletes all contact data of other peers.
- We assume a DHT model like in KAD [SENB07] where peers which publish data are responsible for refreshing this data themselves, i.e., replica nodes do not re-publish data among each other, in particular when some of them leave the network, or new nodes close to the key of the data enter the network. As we discuss later in Section 3, this model is efficient to counter churn attacks.
- We assume that routing in the DHT always succeeds. In particular if Alice is looking for the contact data of Bob and there is at least one replica node p_i storing this data, then Alice will be able to reach p_i and find the contact data of Bob.

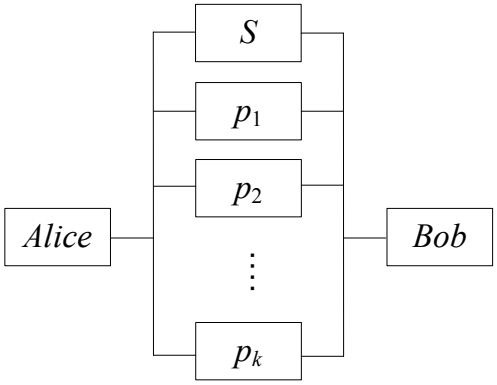


Figure 12: CoSIP reliability model

Figure 12 shows the resulting reliability model under these assumptions. A UA Alice calling Bob needs to reach either the server S or at least one of the storage peers p_i which have stored the contact data of Bob. Thus, assuming uncorrelated failures of the SIP server and the peers in the DHT, the reliability of CoSIP is:

$$R_{CoSIP}(t) = 1 - (1 - R_S(t)) \cdot (1 - R(t))^k \tag{9}$$

where $R_S(t)$ is the reliability of the server, $R(t)$ is the reliability of an arbitrary peer, and k is the number of replica nodes.

As mentioned above, R_S is difficult to estimate without knowing much about the service topology and the reliability of the single units providing the service. Thus, we restrict our reliability analysis to the case where the SIP server fails, which is the aggregated reliability of the k replica nodes storing Bob's contact data:

$$R_{replica}(t) = 1 - (1 - R(t))^k \tag{10}$$

To model the reliability of arbitrary peers $R(t)$, we use the parameters that we deduced from the Weibull fit of the Skype supernode lifetime above (Equation 6). Figure 13 shows the estimated reliability $R_{replica}(t)$ with different number of replica nodes k. Since the Skype model that we have is based on traces of supernodes only, we provide also the estimated reliability based on the KAD model (Equation 7) in Figure 14.

While $R_{replica}(t)$ is slightly higher for the Skype-based model (the difference can be observed, e.g., at the end of the refreshing periods, $1h$, $2h$ and $3h$) in both cases high reliability can be achieved with a quite small number of peers. For example, a reliability of "3 nines" can be achieved with $k = 6$ and a reliability of "5 nines" can be achieved with $k = 10$, in both cases a refreshing period of $1h$.

Pitfalls: The reliability model in Figure 12 is based on some assumptions mentioned above. We now provide some comments on the validity these assumptions.

One assumption that we made for the reliability model is that routing always succeeds. CoSIP uses currently the Kademlia DHT algorithm. Using Kademlia, a peer looking for a data in the network can probe parallel paths to the destination. The Kademlia routing algorithm has proven to be robust [KCTHK09]. Even if a large number of peers leave the network within a short timeframe (up tp 90%), it has been shown that the Kademlia overlay can recover

efficiently [BS07]. Moreover, the connectivity in the overlay can be enhanced if low diameter overlays with degree in $O(\sqrt{n})$ instead of $O(log(n))$ are used as described, e.g., in [TBC+05]. For these reasons, we left the routing reliability out of scope. Nevertheless, successful routing in the overlay requires adequate protection from Sybil and eclipse attacks.

The assumption that all peers are cooperative is not certain. Malicious peers may behave inconsistently, e.g., by responding positively to `STORE` requests but just discarding the data they are supposed to store, or ignoring subsequent `GET` request. Thus, the overlay must allow for sufficient redundancy in routing and storage to counter faulty behavior. Moreover, the overlay must allow for protection against Sybil attacks, which could render redundancy useless.

Finally, we assume uncorrelated failures of the replica nodes. This can notably not be the case if a node launches a successful Sybil attack. A host node may send a `STORE` request to k virtual nodes, thinking that it benefits from high reliability while a malicious node may be running $j \leq k$ nodes, thus invalidating the reliability assumptions of the honest node. Therefore, this is another reason why protection against Sybil attacks.

To summarize, the reliability and security of CoSIP are strongly interrelated. This emphasizes the relationship between reliability and security in the overall context of resilience. The security issues of CoSIP are discussed in Section 3.

2.4.3 Conclusions:

Concluding this section on the reliability of CoSIP, we used Skype traces collected by Guha et al. [GDJ06] to develop a model for the reliability of CoSIP. In contrast to previous models of the Skype supernode lifetime, our analysis shows that the supernode lifetime can be better modeled with a Weibull distribution than with power-law distribution.

CoSIP reliability models based on KAD and Skype indicate that the reliability of the a reliability of "3 nines" or "5 nines" can be achieved with a small overhead, such as CoSIP server downtimes can be bridged with high probability. Thus, the reliability of CoSIP is expected to be significantly higher than the reliability of the pure server solution. Since failures of different units in a system in practice are often correlated, CoSIP provides the benefit of the geographic

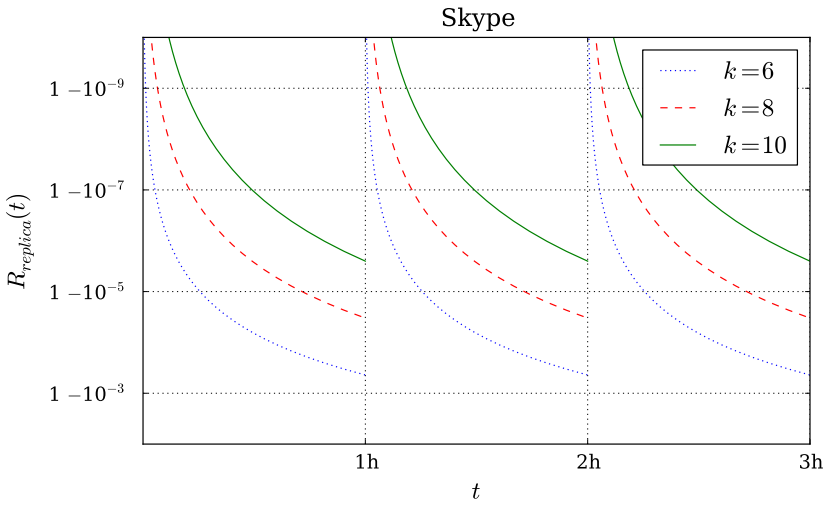


Figure 13: Reliability of k storage peers in parallel using Skype supernodes as a model.

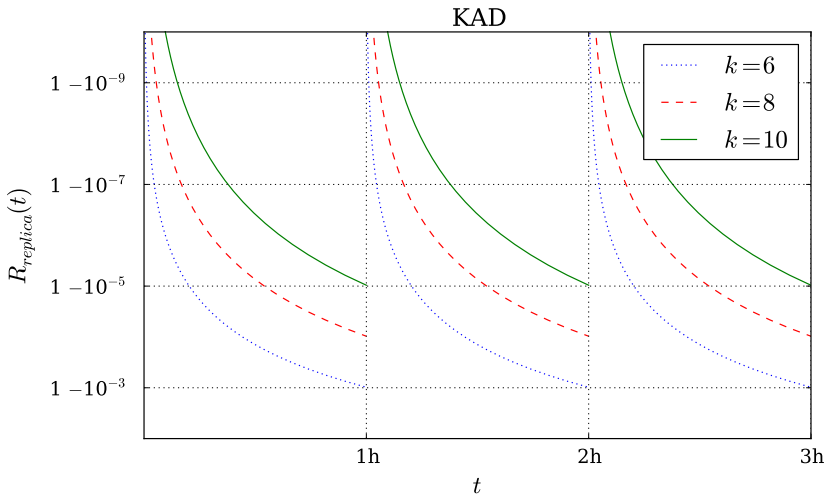


Figure 14: Reliability of k storage peers in parallel using KAD peers as a model.

diversity of the peers (plus server) and potentially diversity in their software and hardware as well. Unless a Caller UA looses connectivity at its access network, it can with very high probability either reach the server or one of the replica nodes storing the contact data of the Callee UA.

2.5 Conclusions

In this section, we presented CoSIP, which is our approach to cope with SIP reliability. CoSIP benefits from the advantages of both server-based and P2P-based SIP networking. It is more likely to survive catastrophic network failures than server-based SIP and provides better security than P2PSIP. We used Skype traces collected by Guha et al. [GDJ06] to develop a model for the reliability of CoSIP. As a side effect of our CoSIP reliability analysis, we observed that the Skype supernode lifetime can be better modeled with a Weibull distribution than with a power-law distribution as in previous work.

CoSIP reliability models based on KAD and Skype indicate that a reliability of "3 nines" or "5 nines" can be achieved with a small overhead, such as CoSIP server downtimes can be bridged with high probability. Thus, the reliability of CoSIP is expected to be significantly higher than the reliability of a pure server solution. CoSIP provides additionally the benefit of geographic diversity of the peers (plus server) and potentially diversity in their software and hardware as well. Unless a Caller UA looses connectivity at its access network, it can with very high probability either reach the server or one of the replica nodes storing the contact data of the Callee UA.

Our prototype CoSIP implementation acts as a local SIP proxy and can be used with standard SIP clients. We successfully validated the functionality of CoSIP on local testbeds as well as on PlanetLab. Security analysis of CoSIP is provided in Section 3.

3 Security in Supervised P2P Networks

Security is inherent in resilience. Thus, a resilient solution for VoIP signaling should allow for the continuity of correct service under normal operation as well as under the presence of malicious and non-malicious faults. In Section 2, we introduced CoSIP, a hybrid architecture for SIP signaling with a server and a P2P network in parallel. Since routing and storage in the P2P network are performed by peers which are not necessarily trustworthy, we need to investigate the security implications of such an architecture. Particularly, we need to address the following questions:

- Does the P2P network introduce new attack vectors?
- Can attackers invalidate the assumptions on the reliability benefits of the P2P network?
- To what extend can CoSIP as a supervised P2P network approach address the security issues inherent in P2P networks and what other security mechanisms are still necessary to provide a resilient SIP signaling solution?

3.1 Security Requirements

We deduce a list of security requirements directly from the security CIA (confidentiality, integrity, availability) model introduced in Thus, a "secure" solution for SIP signaling needs to fulfill the following security requirements:

- *Confidentiality*: absence of unauthorized disclosure of information.
- *Integrity*: absence of improper system alterations.
- *Availability*: readiness for correct service.

This list of security requirements is concise though fulfilling it is not trivial for the following reasons:

- Message confidentiality and integrity of the SIP signaling message and VoIP streams can be achieved upon successful mutual authentication and key establishment between SIP endpoints (Callers and Callee). However, in current SIP signaling solutions SIP endpoints authenticate themselves only towards SIP proxies and registrars. A solution which allows for P2P signaling must allow for mutual end-to-end authentication between SIP endpoints based on their SIP identities.
- Integrity, i.e., absence of improper system alterations in the SIP signaling, is not only about message integrity. It requires also efficient mitigation of SPIT phone calls. Note that a SIP message can be correct while the call is actually illegitimate. Thus, message integrity is not sufficient to mitigate SPIT.

  In server-based SIP, the number of `INVITE` messages from a UA can be compared with a threshold value to detect if the UA is trying to initiate too many phone calls. Other more sophisticated SPIT detection and prevention can be deployed as well [Nic06]. If phone calls are initiated at a P2P basis, it is very difficult to detect such a malicious behavior.

- Confidentiality is not only about encrypting the SIP signaling messages and VoIP streams. It must be assured that no private information about the user is revealed, e.g., location and social interaction in terms of with whom she is communicating. Thus, confidentiality includes also *privacy*.

  Using server-based signaling solutions, only the entities from the SIP infrastructure (proxies and registrars) involved in the signaling dispose of information about the users such as their location and their social interaction. The use of a P2P network for SIP signaling introduces new attack vectors if user location and information about her social interaction are available for arbitrary peers. Given the complexity of this issue, a separate section is dedicated to privacy in CoSIP and P2PSIP. where a solution is provided in details and extensively evaluated.

- The third security requirement is availability. Availability of the SIP signaling using the P2P overlay requires the integrity and availability of:
  - Overlay signaling, i.e., messages for DHT storage, DHT lookup and overlay maintenance,
  - DHT content.

In order to provide a systematic approach to address these security requirements, a threat model is required. This is the subject of the next section.

Before we move forward with the threat analysis, we would like to mention for completeness that there exist potentially more security requirements in a SIP network which are not addressed in this section, e.g., accounting and lawful interception.

3.2 Threat Model

A common threat model for the security analysis of network protocols is the *Dolev-Yao* threat model [DY81]. We first introduce the Dolev-Yao threat model. Then, we introduce a slightly different variant which we will use for our threat analysis.

According to the Dolev-Yao threat model, an attacker Malice

- Can obtain any message passing through the network
- Is a legitimate user of the network, and thus in particular can initiate a conversation with any other user
- Will have the opportunity to become a receiver to any other entity in the network
- Can send messages to any entity by impersonating any other entity in the network. In the context of CoSIP, this means messages either at the overlay layer or the application layer (SIP).

One can think of any message sent to the network as being sent to Malice for her disposal; and any message received from the network as being received from Malice after her disposal.

This assumption is particularly useful in the context of P2P networks since overlay routing and storage are performed by the network participants. An attacker can be another peer in the overlay. Thus, we need to cope with attacks where peers along the overlay routing path may tamper with messages, inject or discard messages. Discarding is notably critical since we need

to consider the free-riding problem where peers use the resources in the P2P network but do not contribute any resources. Thus, they can just discard any message they are supposed to forward. As for storage, one may think of a stored data item in the DHT as a message that is waiting for delivery. Thus, peers storing the data may tamper, inject or discard stored data.

Attacker's Scope Limitations Note that the Dolev-Yao threat model does particularly not deal with attackers which have access to the memory of other network participants, e.g., due to a security flaw in the operating system. These attacks could be used to acquire access, e.g., to encryption keys which are supposed to be secret.

In the context of SIP and P2P networks, this restriction means also that attacks that target security flaws in the implementation of SIP UAs or the P2P protocol are out of scope in our threat analysis.

Weakening The Dolev-Yao Threat Model Albeit an attacker does not have access to the memory of other network participants, the Dolev-Yao threat model attacker is still very strong. Notably, if an attacker controls all communication between two entities A and B, it is not possible to mitigate threats where the attacker just discards all communication. Such an attack would result into the complete loss of SIP signaling availability on top of the overlay. Instead of using an omnipresent attacker, we assume that an arbitrary node in the P2P network involved in forwarding a message from A to B or storing a data item m is malicious with probability p.

3.3 Mechanisms

In this section, we describe mechanisms for addressing the security requirements mentioned in Section 3.1 and according to the threat model introduced in Section 3.2.

3.3.1 Secure Node ID Assignment

The first fundamental tool required is *secure node ID assignment*. A malicious node may generate as many fake node identities in the overlay as possible to increase the probability that she will be responsible for forwarding a message or storing a data item. This malicious behavior has been denoted by *Sybil attack* [8]. Already in the original paper on Sybil attacks by Douceur [Dou02], it was observed that "without a logically centralized authority, Sybil attacks are always possible except under extreme and unrealistic assumptions of resource parity and coordination among entities."

Therefore, a central authority is required in the network to provide verifiable IDs at the overlay layer. We use the CoSIP server as a central authority. It generates a node ID for each node upon successful user registration and embeds the node ID within an X.509 certificate. The CoSIP server needs to keep track of which user has already received which node IDs, in order to limit the number of legitimate node IDs per user. Peers use the X.509 certificates to mutually authenticate each other at the overlay layer, prove the correctness of their respective node IDs, and establish keys for integrity-protection of the overlay signaling.

[8] John R. Douceur [Dou02] was the first who introduced the name "Sybil attack" after a novel about a woman whose name was Sybil [Sch95]. She suffered under multiple personalities. Her mind broke apart and compartmentalized her personality to fifteen other "selves".

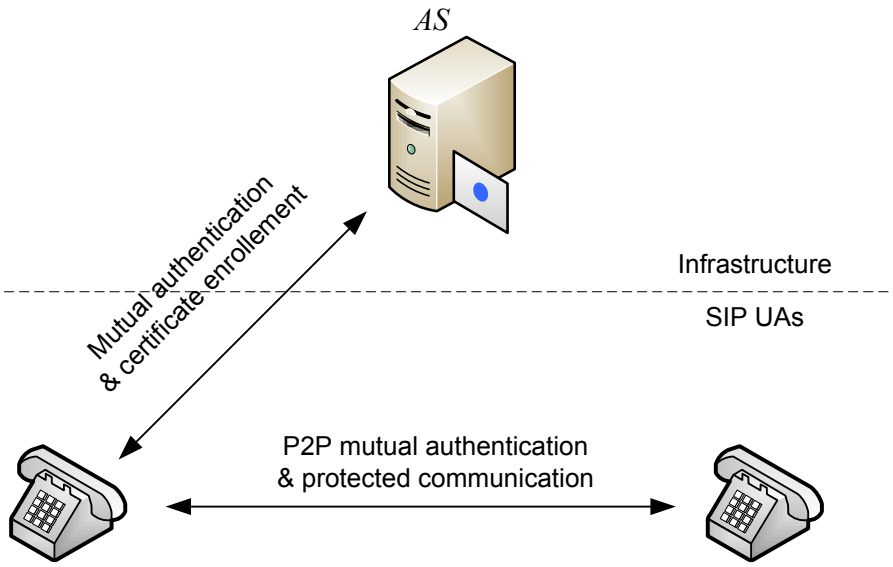


Figure 15: P2SIP Identity Management

Moreover, in Section 3.1, we mentioned that a solution which allows for end-to-end SIP signaling must allow for mutual end-to-end authentication between SIP endpoints. Thus, the CoSIP server provides UAs with verifiable identities at the application layer as well, in form of X.509 certificate including the SIP identity. SIP UAs use these certificates to mutually authenticate each other and establish keys for encryption and integrity-protection of the SIP signaling and subsequent multimedia streams in an end-to-end manner. Moreover, the contact data of a user Alice stored in a DHT can be cryptographically signed by Alice.

Authentication Protocol The two verifiable identities required by a UA are provided by the CoSIP server upon successful user registration. An authentication protocol which runs between a CoSIP endpoint and the CoSIP server provides this functionality. The authentication protocol assumes that the CoSIP Server, or Authentication Server AS, shares a long-term pre-shared key $PSK_{AS,A}$ with UA A. The pre-shared key can be a user password, or a high entropy key stored in the (U)SIM card of the user's smart phone. After successful user authentication, the AS generates the node ID a. It provides the UA with a certificate which binds the node ID a to the public key K_a and a certificate which binds the user's public key $+K_A$ to her SIP URI A.

Figure 15 outlines the CoSIP setup with the AS and the UAs communicating to each other. In fact, this figure shows the difference between the CoSIP authentication protocol and Identity Management (IdM) protocols [PW03]. In IdM protocols, clients acquire credentials from an Identity server to authenticate themselves towards other servers. In the CoSIP case, SIP UAs acquire credentials from the AS (the overlay supervisor) to authenticate themselves to each other.

In the first step of the protocol, client A and AS perform a TLS handshake where the AS is authenticated using a certificate. The server certificate needs to be signed by a common trust anchor, e.g., a root CA which is trusted by both A and AS. Symmetric keys are generated as as side effect of the TLS handshake on both sides for encryption and integrity protection of

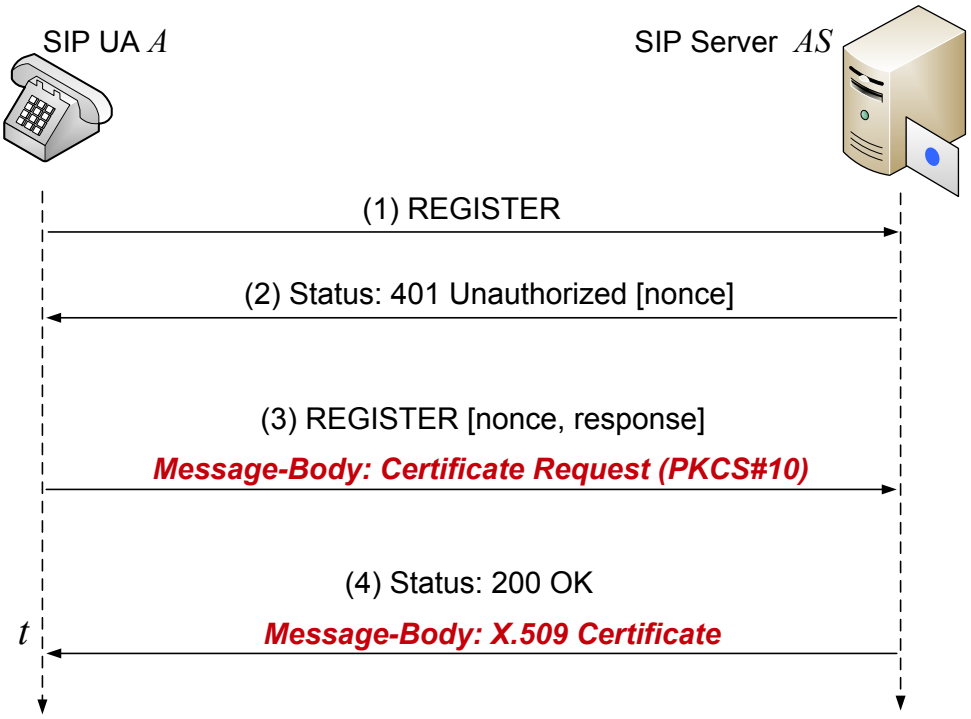


Figure 16: CoSIP registration

the subsequent messages 1 to 4 in message flow (11) below.

Let r be a random number freshly generated by AS at each protocol run. Further, let $Cert_{AS}(e, +K_e)$ denote a digital certificate for e issued by AS. We abstract from details, e.g., certificate validity intervals. h is a cryptographic hash function, and '|' the concatenation operator. A successful authentication protocol run then looks as follows:

$$\begin{array}{lrcl} 1: & A \rightarrow AS & : & Identity\_req \\ 2: & A \leftarrow AS & : & r \\ 3: & A \rightarrow AS & : & h(\,A \,|\, PSK_{A,AS} \,|\, r\,), +K_A, +K_a \\ 4: & A \leftarrow AS & : & Cert_{AS}(A, +K_A), Cert_{AS}(A, +K_A) \end{array} \quad (11)$$

A successful run of the protocol enables A to authenticate herself towards the server, and then acquire the two required verifiable identities.

Implementation We implemented the authentication protocol as an extension of our CoSIP implementation (See Section 2). In contrast to legacy SIP where a `REGISTER` message does not carry a message body, a certificate request is carried as message body. Upon successful authentication, the UA receives a X.509 certificate in the 200 `OK` message. We extended the SER authentication module as well as our CoSIP proxy implementation for this purpose. Figure 16 highlights the differences between a legacy SIP user registration message flow and CoSIP user registration message flow.

Listings 1 and 2 (Below at the end of this section) show the `REGISTER` message and the 200

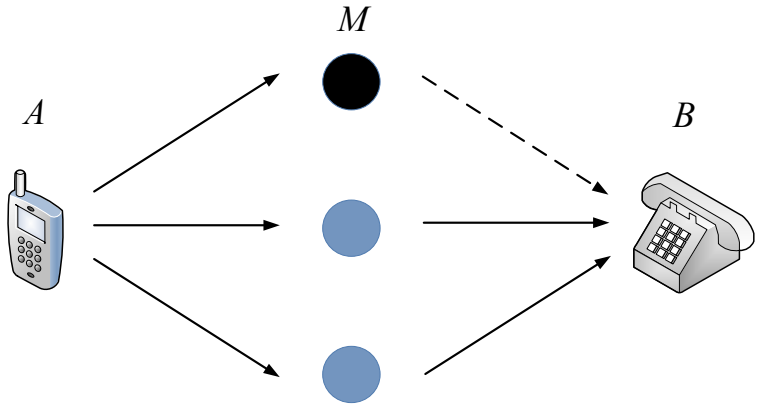


Figure 17: Parallel routing in DHTs with an attacker on one of the paths.

`OK` message from traces of a successful authentication protocol run using our implementation. These correspond to messages no. 3 and no.4 from message flow (11) above. The `REGISTER` message includes the Base64-encoded certificate request. The Authorization Header includes the challenge from the server (*nonce*) and the corresponding response which has been computed by the UA based on the preshared key. The `Content-Type` is `application/pkcs10`, i.e., a certificate request [Tur10]. `Content-Length` and `Content-Transfer-Encoding` are additional headers required to process the message body which otherwise does not exist in legacy SIP registration. The `200 OK` response from the server which includes the X.509 certificate.

3.3.2 Resilient Routing

Second, we address the problem of maliciously discarding overlay lookup messages.

Parallel Lookups Figure 17 shows how a lookup message is forwarded via one hop in parallel from A to B. Let k be the number of parallel nodes between A and B and p is the probability that an arbitrary peers is malicious. Thus,

$$P[routingsuccess] = 1 - p^k \tag{12}$$

For example, if A sends $k = 3$ parallel requests to B, and $p = 0.2$, this results into a routing success probability of 0.992. Since this success probability might still be not sufficient, A may probe different paths in a wider scope if routing fails. This is the case, e.g., for Kademlia [MM02] where the number of parallel lookup requests is increased to the size of the k-buckets, e.g., $k = 8$, or $k = 20$.

Iterative vs. Recursive Routing Using *recursive overlay* routing, a message is forwarded from a source A to a destination D hop by hop, e.g., via nodes B and C as follows:

$$A \rightarrow B \rightarrow C \rightarrow D$$

The response is either sent directly to A or hop-by-hop backwards taking the same path:

$$D \rightarrow C \rightarrow B \rightarrow A$$

In the latter case, the routing is *symmetric recursive routing*. Using *iterative routing*, the first hop on the path B does not forward the lookup message to C. Instead, it responds to A providing the contact data (IP and port) of C. Then, A contacts C directly and so on.

Figure 18 shows how a message is forwarded from A to B recursively.

Let k be the number of hops between A and B. Thus,

$$P[routingsuccess] = (1-p)^k \tag{13}$$

Taking the Chord DHT algorithm [SMK+01] as an example, the average path length between two nodes has been proven to be $log_2(n)/2$ [LKRG03]. Thus, let's assume $k = log_2(n)/2$. Figure 19 shows the routing success probability for different values of p.

As we observe in Figure 19, the routing success probability drops quickly by increasing n. A routing availability of "5 nines" is far from being reached.

A may try another parallel path as described above. However, there are still two problems faced with recursive routing:

Path diversity: Since A has only contact to the first hop on the path, she can not identify which peer is to blame if a message gets lost. She does not even know the other peers on the path. Thus, she can not guarantee that parallel lookups do not converge towards an attacker on the path as shown in Figure 20. In other words, A is not able to identity multiple node-disjoint paths to guarantee that parallel lookups would bring any improvement.

Note that the same problem is encountered in case peers are not malicious but just go offline without notifying their neighbor peers.

Lookup latency: Since A can only roughly estimate the path length and does not have any information about the round trip time between every two hops on the path, it is very difficult to estimate the overall lookup latency and to setup an appropriate timeout interval to re-initiate the lookup. This leads to a waste of time.

To conclude, by simply discarding all signaling messages, an attacker can cause a serious harm to the availability of the overlay routing. The consequences are higher lookup latency and a significant degradation of lookup availability. The solution is twofold:

- Parallel lookups to increase the probability to successfully route beyond faulty nodes.
- Iterative routing to better control the lookup routing progress. This reduces the required lookup request timeouts, and allows for detecting faulty nodes more efficiently and routing beyond them.

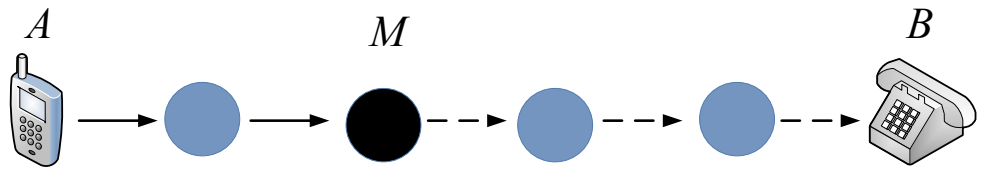


Figure 18: Recursive routing in DHTs with an attacker on the path.

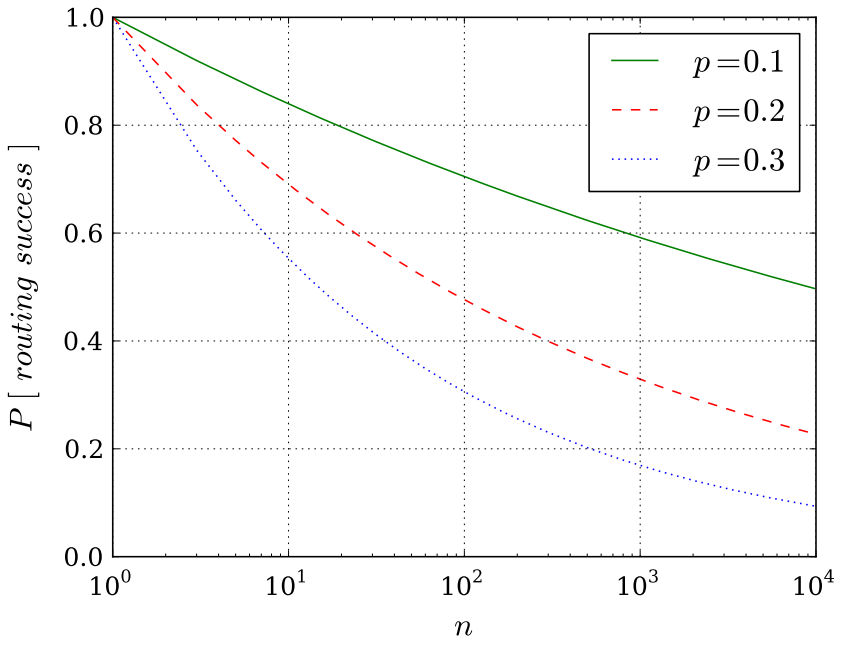


Figure 19: Routing success probability using recursive routing with different attacker ratios p.

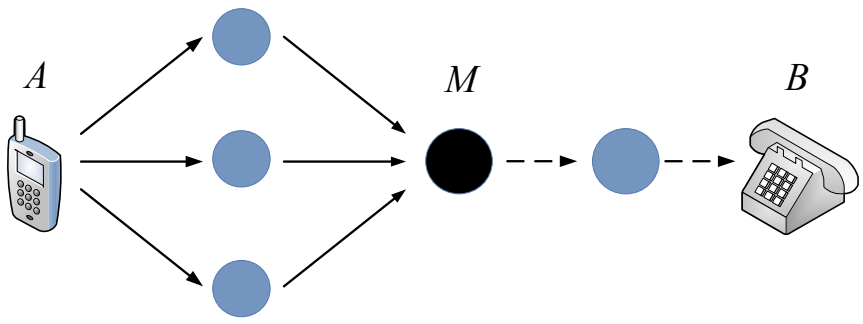


Figure 20: Parallel recursive routing in DHTs with an attacker on the path.

3.3.3 Data Replication

For the purpose of completeness, we mention data replication as a mechanism for addressing the threat model from Section 3.2 since attackers may simply discard data items instead of storing them.

As discussed in Section 2, data replication is inherent in P2P networks to provide high reliability. If A publishes her contact data at k nodes and $j \leq k$ nodes among the replica nodes are malicious, this reduces the availability of A's contact data in the DHT. However, given that secure node ID assignment provides protection against Sybil attacks, we assume that a malicious peer can not control more than one peer in the overlay. Thus, the incentives for a malicious peer to discard data are minimal for the following reasons:

- The probability that discarding the data would successfully lead to the unavailability of the contact data of A is very low, since there are sufficiently other replica nodes in the DHT that would be able to deliver it.
- Storing the contact data of A does not require a considerable amount of resources.

3.4 Evaluation

In this section, we provide an attack taxonomy to evaluate to what extend the security mechanisms described in Section 3.3 do successfully remediate attacks on P2P networks, notably in the context of CoSIP. The attack taxonomy is classified in a layered approach into:

i) Attacks on the overlay,

ii) Attacks on the DHT,

iii) Attacks on the application.

The impact of attacks on one of these layers may propagate to upper layers. For example, attacks targeting the overlay routing availability may lead to the failure of storing or retrieving content in or from the DHT. This in turn may lead to the failure of SIP user registration in the DHT, or SIP session establishment [9].

Some attacks are valid at several layers. For example, flooding attacks may be performed at the overlay layer by generating overlay signaling messages, or at the application layer by generating SIP signaling messages. We will mention attacks which are valid at different layers at each of the corresponding layers for completeness.

3.4.1 Attacks on the Overlay

Message Manipulation An attacker may tamper with, insert or discard messages it is forwarding in the overlay.

[9] This is, in fact, similar to the propagation of the impact of attacks in the Internet layers. For example, attacks on a cable may lead to the unavailability of IP connectivity between two IP hops. A DoS attack on a IP router may lead to the lack of availability of web content

Impact: Loss of overlay signaling integrity. Loss of overlay signaling availability if overlay messages are discarded.

Countermeasures:

- Integrity protection of overlay signaling to detect forged messages.
- Parallel and iterative overlay routing to counter message discarding.

Request Hijacking Request hijacking is a special case of message manipulation. An attacker intercepts a request from peer A and sends a forged response. Additional mechanisms may be required by the attacker to prevent the correct responder from sending a response or to prevent the correct response from reaching the request initiator.

Impact: Loss of overlay signaling integrity.

Countermeasures: Integrity protection of overlay signaling. Parallel routing to counter illegal message interception.

Peer Impersonation A malicious node uses a fake ID in the P2P network or misuses the ID of another peer.

Impact: Loss of overlay signaling integrity.

Countermeasures: Cryptographically verifiable peer IDs. Integrity protection of overlay signaling.

Invalid Message Forwarding A malicious node may forward overlay messages to an invalid node, non-existing nodes or existing but random nodes, or simply discard the overlay messages.

Impact: Higher overlay routing latency. Degradation of overlay routing availability.

Countermeasures: Parallel and iterative overlay routing.

Propagating Wrong Routing Tables A malicious node may even propagate wrong routing information and force other honest nodes to forward overlay messages incorrectly.

Impact: Incorrect routing tables result into high overlay routing latency and degradation of overlay routing availability.

Countermeasures:

- Routing table entries should be validated before they are adopted. This can be performed by the following mechanisms:

  - Each routing entry propagated should include the node ID, location (IP and port) and certificate. This prevents attacks where malicious nodes propagate routing entries with non-existing nodes.
  - Additionally, a node must reply to an overlay layer `Ping` message before it is adopted in the routing table. This prevents attacks where malicious nodes propagate routing entries with existing but offline nodes.
- Rejection of faulty or stale routing tables entries.
- Parallel and iterative overlay routing to efficiently route beyond faulty or stale routing table entries.

Bootstrap Attacks If a honest node contacts a malicious peer to join the P2P network, the malicious peer may provide the honest peer with wrong overlay information.

Impact: The honest node will join a parallel overlay network ⇒ Loss of availability.

Countermeasures: Node certificates must include an identifier of the overlay, such as new nodes can verify that they are in the correct overlay. For example, in X.509 certificates the node ID can be in the Subject Common Name (CN) and the overlay name in the Organization (O) or Organizational Unit (OU).

Chosen-Location Attacks The attacker chooses an ID in the overlay such as it it located in a strategically good location in the overlay, e.g., in the vicinity of their targeted victim peer. A chosen-location attack does not cause harm by itself. It rather increases the success probability of subsequent attacks.

Impact: By choosing a location in the vicinity of a victim A, the attacker M increases the probability to be responsible for forwarding messages from or to A. This may lead to loss of overly routing integrity or availability.

Countermeasures: A peer must not be able to choose its own ID or even choose a preferred sector of the ID space. The overlay supervisor is responsible for generating node IDs.

Flooding Attacks An attacker may launch a DoS attack against one or more nodes in the overlay by flooding them with a large number of overlay messages, e.g., routing requests.

Impact: Increased CPU, memory and bandwidth usage at the victim peer ⇒ Resource depletion. Degradation of overlay availability.

Countermeasures: Typical mechanisms against flooding attacks. For example, cookies [Lem02] or puzzles [ANL01] which can be generated by the victim and verified with low CPU resources. They may help against attackers with spoofed IP addresses. However, network congestion may not be prohibited successfully. The benefits of cookies and puzzles in case of large scale distributed DoS attacks are limited.

DoS Attack Amplification by Misusing Recursive Routing Using recursive routing; each lookup message in the network generates $O(log\, n)$ messages. This amplifies flooding attacks.

Impact: Increased CPU, memory and bandwidth usage in the P2P network ⇒ Resource depletion. Degradation of availability.

Countermeasures: Iterative routing ⇒ Each message generated by an attacker generates a single response.

DoS Attack Amplification based on Churn Enforcement Some DHT algorithms, e.g., Chord [SMK+01] and Pastry [RD01a] require adapting overlay routing tables each time a new node joins or leaves the network. Malicious nodes may misuse this to generate a heavy load in the network by joining and leaving the network frequently. This can be misused to amplify DoS attacks.

Impact: Increased CPU and bandwidth usage in the P2P network ⇒ Resource depletion. Degradation of availability.

Countermeasures: Dampening the impact of joins and leaves. New joining nodes should not cause immediate major changes in the routing tables of other peers.

Sybil Attacks Sybil attacks [Dou02] are a generalization of peer impersonation attacks. A malicious node joins the P2P network with multiple fake identities.

Impact: Other nodes believe that they are interacting with different nodes, while they are actually interacting with the same node. A malicious node performing a Sybil attack may be able to gain control over a large part of the P2P network. A Sybil attack does not cause harm by itself. It rather increases the success probability of subsequent attacks.

A Sybil attack increases the probability that an attacker disposes of a message sent from A to B and thus increases the ability to manipulate it.

Countermeasures: Verifiable peer IDs signed by a central authority. The central authority needs to control the number of assigned verifiable peer IDs per user.

Node Eclipse Attacks Node eclipse attacks [SM02a] are a special case of message and data manipulation attacks. They are called after the term 'eclipse' in astronomy which means that an object moves into the shadow of another. In the context of P2P networks, a node eclipse attack is an attack where a node is shielded by an attacker.

Impact: A node eclipse attack on a node renders all communication from and to that node controlled by the attacker. This leads to loss of integrity and availability of the overlay signaling.

The basis for a successful eclipse attack may be a combination of a Sybil attack and chosen-location attacks.

Countermeasures:

- Prevent Sybil and chosen-location attacks.
- Parallel and iterative overlay routing to increase the probability to route beyond the eclipse attacker.

Inconsistent Behavior Malicious peers may behave inconsistently in order to not reveal their actual intention and depending on the situation. Any of the attacks on the overlay above may be combined with inconsistent behavior. For example, they may occasionally discard overlay messages they are supposed to forward. A malicious node M may always respond to keepalive messages from a honest peer A in order to convince A to keep M in her overlay routing table. However, M may still discard incoming overlay messages which should be forwarded to A.

Impact: Higher overlay routing latency and potentially lack of overlay routing availability.

Countermeasures: Parallel and iterative overlay routing to increase the probability to route beyond the inconsistent attacker.

3.4.2 Attacks on the DHT

Content Manipulation An attacker may tamper with, insert or delete data items in the DHT. A malicious peer may delete a data item locally or may send a `DELETE` request to other peers storing the data item.

Impact: Loss of DHT content integrity or DHT content availability.

Countermeasures: Integrity protection of data stored in the DHT. Data replication to alleviate the impact of data discarding. DHT content must be protected from illegitimate delete operations by verifying that the peer sending the `DELETE` request is the verifiable "owner" of the data item.

Chosen-Location Attacks Chosen-location attacks may target a victim node (See Section 3.4.1 above) or DHT content, e.g., the contact data of a SIP UA.

Impact: By choosing a location in the vicinity of the key for a data item m, the attacker M increases the probability to be responsible for m. This allows for DHT content manipulation. This may lead to loss of DHT content routing integrity or availability.

Countermeasures: Same as countermeasures for chosen-location attacks at the overlay layer (See Section 3.4.1).

DoS Attack Amplification based on Churn Enforcement Churn may cause not only immediate changes in the routing tables as mentioned in Section 3.4.1. For example, Chord [SMK+01] and Pastry [RD01a] require also reshuffling data each time a new node joins or leaves the network. This can be misused to amplify DoS attacks.

Impact: Increased CPU and bandwidth usage in the P2P network ⇒ Resource depletion. Degradation of availability.

Countermeasures: Dampening the impact of joins and leaves. Stored data should not be reshuffled. Instead, the data publisher should be responsible for refreshing the data by himself and guaranteeing that the right replica nodes dispose of it.

Sybil Attacks Sybil attacks may target the DHT content as well.

Impact: A Sybil attack increases the probability that the attacker will be responsible for storing a data item. This allows for the manipulation of that data item. Furthermore, the attacker will be able to control to whom it wants to provide the data item. This leads to lack of DHT content integrity and availability.

Countermeasures: Same as Sybil attacks on the overlay. See Section 3.4.1.

Content Eclipse Attacks A content eclipse attack is an attack where a data item in the DHT is shielded by an attacker.

Impact: A content eclipse attacks renders the data item unavailable for lookup from other peers which, i.e., loss of content availability.

Countermeasures:

- Prevent Sybil and chosen-location attacks.
- Parallel and iterative lookup to increase the probability to reach one of the replica nodes.
- Sufficient data replication by the publisher to reduce the success probability of a content eclipse attack.

Inconsistent Behavior Malicious peers may behave inconsistently in DHT as well. For example, they may discard data they are supposed to store while still responding positively to a `STORE` RPC.

Impact: Lack of DHT content availability.

Countermeasures: Sufficient replication.

3.4.3 Attacks on the Application

UA Impersonation A malicious UA uses a fake SIP URI or steals the URI of another UA.

Impact: Loss of SIP signaling integrity.

Countermeasures: Verifiable SIP URIs. Integrity protection of SIP signaling.

Spam over IP Telephony (SPIT) *Impact:* User annoyance.

Countermeasures: Accept phone calls only from known UAs. Obviously this countermeasure reduces the application availability though.

SIP Flooding Attacks SIP flooding attacks have the same properties in terms of impact and countermeasures as the flooding attacks at the overlay layer (See Section 3.4.1).

Eavesdropping & Traffic Analysis A malicious node may passively record activities of other users or peers in the network. A malicious node may use this information to build profiles with locations and social interaction of its neighbors, e.g., when and whom they are calling.

Impact: Loss of confidentiality. Loss of privacy.

Countermeasures: Encryption of SIP and overlay signaling. Deployment of anonymization techniques.

3.5 Conclusions

CoSIP uses a P2P network as a backup for session establishment. Thus, it is critical to prevent attacks which aim at invalidating the reliability benefits of the additional P2P network. These attacks may target the overlay routing or the DHT content. Being a supervised P2P network approach, CoSIP addresses the security issues in P2P partially by providing secure node ID assignment. This addresses the issues of Sybil attacks, chosen-location attacks and eclipse attacks. Moreover, verifiable SIP URIs allow for P2P mutual authentication and establishment of security context at the application layer; and integrity-protection of DHT content by digital signatures.

However, attacks on the P2P network can not simply be resolved by adding some cryptography. Notably, even if all overlay messages are integrity-protected, a malicious node responsible for forwarding a message may forward it to the wrong peer, or may simply discard it. A malicious node responsible for storing a data item may simply discard it or refuse to deliver it. Therefore, the resilience of P2P networks require additional mechanisms such as redundancy in routing (parallel lookups) and storage (replica nodes). Moreover, iterative routing allows for the limitation of the impact of attacks on the routing and prevent the amplification of flooding attacks.

Listing 1: CoSIP REGISTER message with a certificate request in the message body.

```
1  REGISTER sip:192.168.1.13 SIP/2.0
2  From: "alice" <sip:alice@192.168.1.13>
3  To: "alice" <sip:alice@192.168.1.13>
4  Via: SIP/2.0/UDP 192.168.1.13:10000;branch=z9hG4bK4D2753BD
5  Call-ID: 1795242434@192.168.1.13
6  CSeq: 7019 REGISTER
7  Contact: "alice" <sip:alice@192.168.1.13:10000;transport=udp>;methods="
       INVITE,MESSAGE,INFO,SUBSCRIBE,OPTIONS,BYE,CANCEL,NOTIFY,ACK,REFER"
8  Authorization: Digest username="alice", realm="192.168.1.13", nonce="4900
       d5c0000000001a93b10143b4d83179311d18f24f53c8", uri="sip:192.168.1.13",
       cnonce="abcdefghi", nc=00000001, response="
       b00fae2e774203be0de6edc60ee93025", opaque="", algorithm="MD5"
9  Expires: 900
10 Content-Length: 525
11 Content-Type: application/pkcs10
12 Content-Transfer-Encoding: base64
13 User-Agent: kphone/4.2
14 Event: registration
15 Allow-Events: presence
16
17 -----BEGIN CERTIFICATE REQUEST-----
18 MIIBTDCBtgIBADANMQswCQYDVQQGEwJVUzCBnzANBgkqhkiG9w0BAQEFAAOBjQAw
19 gYkCgYEA4ygxLtsL3PRIAnblaBoD1MIEVCtuQfHB51rXngeCc6Kioyly9tg8DBXw
20 ZP0/z5NRu+SKyoC+9IkI2eGiSWZ27ve50I5VsKhLSzgg36UJ6KoheDJkilrxZ/2R
21 xmoVe6zx2R86VSRBYVat6dxpUjwUw4PgMW+qeVn9WvHUNd1FUk8CAwEAAaAAMA0G
22 CSqGSIb3DQEBBQUAA4GBAJ7VQ+xUb99U069mRXRonnIxO0236D3D/+2FhuXalkBo
23 LRKQLXMXeiAJjzjWvIQX9uT0F/7X4UO8xHTjIM8DFWyr8Re+YZ5oeDzi3hVAck4p
24 sDvP9Cu6ICNHrmUV93uUf9ed7o3Dk/wPKvNBqriKiQGDnLWFtJ677Qp2ikCNcUg/
25 -----END CERTIFICATE REQUEST-----
```

Unfortunately, the P2P network does introduce new attacks as well. Two issues are difficult to solve, namely SPIT and attacks on privacy. SPIT attacks are easier in a P2P mode. One way to counter this problem is to accept phone calls only from known UAs or to exploit knowledge from social networks to evaluate a SPIT score for incoming calls. However, anti-SPIT mechanisms at the infrastructure are expected to perform better than on a P2P basis. This confirms the fundamental design decision of CoSIP to use the P2P network only parallel to the infrastructure and not as the single solution.

Attacks on privacy result from the fact that user location is stored in the DHT and available for all network participants. Moreover, session establishment involves arbitrary peers which are able to deduce that some users have a social interaction. A solution for the privacy issues is provided and extensively evaluated in [FENH10].

Finally, we would like to emphasize the strong interrelationship between reliability and security. As observed, an attacker can invalidate the reliability estimation of the P2P network that we computed in the reliability analysis in Section 2. For example, storing an object at k replica nodes provides redundancy, but becomes useless if all k replica nodes are controlled by a single malicious node launching a Sybil attack. The mechanisms proposed for addressing the security issues are typical mechanisms for enhancing the reliability of a system or a network. For example, data replication or multipath (parallel) routing [LCR+07]. This observation confirms the validity of the ResumeNet strategy to consider both disciplines in joint efforts and the resilience of a system in a holistic approach.

Listing 2: CoSIP 200 OK message with a X.509 certificate in the message body.

```
SIP/2.0 200 OK
From: "alice" <sip:alice@192.168.1.13>
To: "alice" <sip:alice@192.168.1.13>;tag=650b16d746387f5eaabb6f44b302fd3d.4
    fc2
Via: SIP/2.0/UDP 192.168.1.13:10000;branch=z9hG4bK4D2753BD
Call-ID: 1795242434@192.168.1.13
CSeq: 7019 REGISTER
Content-Type: application/pkix-cert
Content-Transfer-Encoding: base64
Server: Kamailio (1.4.1-notls (i386/linux))
Content-Length: 603

-----BEGIN CERTIFICATE-----
MIIBkDCB+gIJAPBYEyxmEWKiMA0GCSqGSIb3DQEBBQUAMA0xCzAJBgNVBAYTAlVT
MB4XDTA4MTAyMzE5NDYyOFoXDTA4MTEwMjE5NDYyOFowDTELMAkGA1UEBhMCVVMw
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMIufUsxPnVo/uUhe5smeiX2YZw8
s1dy1Ll8WW2cVPxM3gZ+Tz3O+5XLbRh3NrFkZs8IbAt3T7uur5BqsdAJA0AcMLuR
qRs9raJ2IrRaofhmpfP32Mo0doFwxabGq5sXswiUGtuUYoBI6IiBJEMSseohLaic
dyY5k5F1K5nBo/jvAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEALzyYltT/+15XpA7+
/86P9u10LyLi31DyZo/CeKFNW7fLivMoVMpOYaVF4TL7ybVPv4HQ/zAhCnBcxHgk
6W3yxRHsZUH7/WKOtwavvvRuu9Qf6dGbiZXtpTPLpC+0aLHNVISfOV7Bn78Ybita
+QBonxxCINEDoPlepaWoo/cwApE=
-----END CERTIFICATE-----
```

4 Related Work

The goals of CoSIP are to improve *i)* reliability compared to server-based SIP and *ii)* security compared to P2PSIP. Thus, we describe related work in these areas.

P2PSIP P2PSIP has been initially proposed by [BLJ05] and [SS04] and raised much interest and follow up work. In 2010, at IPTComm, the conference dedicated to IP telephony[10], there was a separate session for P2P telephony. REsource LOcation And Discovery (RELOAD) [JLR+10b] is the IETF base protocol for P2PSIP. It is a generic P2P protocol. It allows for different overlay algorithms to be plugged in and different usages. This means it is a P2P protocol not not limited to SIP. The Internet draft [JLR+10a] describes SIP usage for RELOAD.

The differences between CoSIP and P2PSIP are as follows: P2PSIP is intended to function without central entities. Discussions at the IETF where the author of this document was involved[11] argued for the necessity of a central authority in order to counter Sybil attacks and chosen-location attacks. Thus, the deployment of a certificate enrollment server was introduced in the RELOAD draft. Nevertheless, the functionality of this central entity has not been further specified. The intention is to keep it as simple as possible in order to save the infrastructure costs that would be otherwise required.

In contrast to P2PSIP, in CoSIP user registration and session establishment run in parallel at the server and the DHT for the following reasons:

- Unless the network is sufficiently small such as the establishment of a security context could be performed manually, a central authority is required anyway. A fundamental result on authentication protocols by Boyd [BM03] says that authentication can either be based on an already existing context or on a trusted third party which is a central authority.
- Session establishment does not generate a high workload on the SIP infrastructure as it is commonly believed. In his dissertation, Singh [Sin06] showed that a cluster of six commodity PCs can support 10 million busy hour call attempts (BHCA), and 10 million users, and thus, exceeds the performance of a typical class-5 PSTN switch costing millions of dollars.
- Using the SIP infrastructure for session establishment under normal operation provides advantages compared to a pure P2PSIP solution, notably better SPIT prevention.

For these reasons, we came to the conclusion that although the IETF P2PSIP solution RELOAD in the meanwhile does not exclude the usage of a certificate enrollment server, CoSIP remains a superior approach by using the P2P network to improve the reliability of SIP signaling, though in combination with SIP servers.

Skype Skype has become very popular in the last few years. Skype uses a mixture of client/server and P2P technologies. Some tasks, such as lookups, are performed on a P2P basis and others, such as login and PSTN gateways use the centralized servers. Being a hybrid architecture, Skype shares several commonalities with CoSIP. However, the main problem with

[10] http://iptcomm.org/

[11] See, e.g., `http://www.ietf.org/mail-archive/web/p2psip/current/msg03470.html`

Skype is that it uses a proprietary protocol. Our intention is to develop an open protocol for VoIP signaling which can be deployed by different vendors and service providers and interoperate with existing client-server SIP infrastructures. Furthermore, given that Skype is a closed source application, the threat model is different than in CoSIP or P2PSIP. Thus, different mechanisms for security are required in CoSIP or P2PSIP as discussed in Section 3.

CoDNS DNS provides a similar functionality to SIP, since it resolves a Fully Qualified Domain Name (FQDN) to the location of a server (i.e., IP address and optionally port number). There has been several proposals for using a P2P network for DNS signaling [PPPW04, RS04, Mas06, CMM02]. An extensive comparison with these approaches is out of scope for this document. But we would like to mention one of them which is the most similar to CoSIP: The name "CoSIP" is actually inspired by CoDNS [PPPW04]. Using CoDNS, DNS clients send a DNS lookup to another DNS client in another domain when they notice that their DNS server is encountering problems. However, the authors of CoDNS do not cope sufficiently with security issues. They admit that CoDNS can not cope with forged DNS responses. CoSIP on the other hand, can cope with the security issues by having a central authority. User contact records stored in the DHT are integrity-protected (See Section 3). On the other hand, CoDNS could be extended to support signed DNS records based on DNSSEC.

Security in P2P Networks The security issues in P2P networks discussed in Section 3.4 have been initially described in [SM02b]. The P2P research community has then spent a considerable effort on addressing these security issues in a distributed manner [BM07, DLITT05, DH00, WZH05, MGM05, DA06]. However, these solutions can not provide resilience guarantees. One of the main problems is that they do not provide protection against Sybil attacks. For example, [BM07] provides a mechanism to route DHT lookups beyond malicious nodes and show that if 20% of the nodes in the network are malicious, the lookup success rate is at 99.0%. This is obviously far from being sufficient for reliability telephony. First a lookup availability of 99.0% is not sufficient. Second, an attacker with sufficient resources can easily launch a Sybil attack and emulate more than 20% of the nodes in the network.

Further research effort exclude security issues in their work by simply proposing the use of a PKI [RD01b, BLJ05], but do not specify how such a PKI can solve the security problems. Seedorf [See06a] discusses the security issues inherent in P2PSIP. In [See06b] he proposes self-certifying SIP-URIs. A SIP-URI is generated based on a cryptographic hash value of the user's public key. This can be useful to protect the integrity of the DHT content. However, it does not provide any protection against Sybil attacks or attacks on the routing. Moreover, cryptographic SIP-URIs are annoying from usability perspective. In [See08], Seedorf investigates the challenges of lawful interception (LI) in P2PSIP networks, and potential solutions. In [BSE09], the authors investigate a game theoretical approach for the security threats of P2PSIP such as SPIT and attacks on overlay routing.

The IETF P2PSIP base protocol RELOAD specifies the enrollment of certificates from an enrollment server. However, a certificate is used for both user and node authentication. It is well known [FS00] that different keys should be used for different purposes. Thus, in contrast to RELOAD we use different certificates for the two different verifiable identities.

While our analysis of the negative impact of recursive routing on the routing resilience is comprehensible, recursive routing is still considered as an option. Notably, The RELOAD draft specifies the use of symmetric recursive routing. The main motivation behind symmetric recursive routing is that the request initiator does not need to perform the required NAT

traversal procedure [Ros10] to guarantee IP connectivity with each of the nodes on the path during the lookup. This approach may be useful to save traffic and lookup latency if the network is stable and the probability that there are malicious peers on the path is very low. However, unless the P2P network would be a network of servers (in which case NAT traversal would not be a problem) where all nodes are trustworthy, this assumption seem to be unrealistic to us.

The security solutions for CoSIP presented in this section have been designed with a strong focus of the resilience and the reliability of the service provided by the overlay. Thus, although there has been a good deal of work on the topic of the security of P2P networks, we consider the security analysis and mechanisms discussed in this section as distinctive from other work in these areas and appropriate in the context of resilient IP telephony.

5 Conclusions

In this document, we presented CoSIP (Section 2), a supervised P2P network approach to address the reliability issues in the context of VoIP. In CoSIP, SIP endpoints organize themselves in a DHT. User registration occurs in parallel at the SIP server and in the P2P network. In case of server failure, the Caller can still establish phone calls by retrieving the Callee contact data from the DHT. We have shown based on reliability theory and traces from the Skype network that the P2P network can enhance reliability by "3 nines" or "5 nines" with a small number of replica nodes (6 for "3 nines" and 10 for "5 nines"). Therefore, CoSIP server downtimes can be bridged with high probability. Moreover, CoSIP provides the benefit of geographic diversity of the peers (plus server) and potentially diversity in their software and hardware as well. Thus, the reliability of CoSIP is expected to be significantly higher than the reliability of a pure server solution.

On the other hand, SIP endpoints acquire verifiable identities from the CoSIP server (the supervisor) upon successful user authentication. This improves the security compared to pure P2P networks. A SIP UA acquires one verifiable identity at the application layer in form of a SIP URI embedded in an X.509 certificate. In contrast to current SIP networks where the SIP endpoints authenticate themselves only to SIP proxies and registrars, they can use the certificates for P2P authentication among each other. The second verifiable identity issued by the supervisor is used at the overlay layer. It is a node ID chosen by the supervisor and embedded in an X.509 certificate. It allows for proving the correctness of the node ID to other peers in the overlay and provides protection against Sybil, eclipse and chosen-location attacks.

The approach with verifiable IDs is similar to Skype where Skype peers acquire certificates upon successful login [Ber05]. However, Skype uses a proprietary protocol. Our intention is to develop an open protocol for VoIP signaling which can be deployed by different vendors and service providers and interoperate with client-server SIP infrastructures. Moreover, given that it is a closed source application, the threat model is different.

The extensive security threat analysis in Section 3 has shown that verifiable IDs are not sufficient to address the security issues in P2P networks. For example, a malicious peer can still discard lookup messages, or forward them to random peers. Likewise, a malicious peer can still discard data items it is supposed to store or refuse to deliver them upon lookup requests. These attacks result in reducing the availability of the SIP signaling running on top of the overlay. Thus, security in P2P networks needs to be enhanced with redundancy in routing and storage. In particular, parallel lookups increase the probability to successfully route beyond faulty nodes. Iterative lookups allow for efficient detection of faulty nodes leading to a reduction of the lookup latency and an increase of the lookup availability.

The security threat analysis left two security threats unsolved, which result from using the P2P network for VoIP signaling. These threats are SPIT and attacks on privacy, notably location privacy and social interaction privacy. Concepts for SPIT prevention can be deployed only with limitations in a P2P environment. Therefore, it is recommended to allow for P2P signaling in CoSIP only with previously known UAs. Thus, in case of server failure, users can, e.g., still reach their family and friends using the P2P network. On the other hand, this confirms the CoSIP concept where the SIP infrastructure should used for session establishment under normal operation and the P2P network is used for enhancing reliability but not as the only signaling solution. As for privacy, a solution has been provided (attached to D3.1b and meanwhile published in [FENH10]).

References

[3GP08] 3GPP. IP Multimedia Subsystem (IMS); Stage 2. TS 23.228, 3rd Generation Partnership Project (3GPP), September 2008. Available from: `http://www.3gpp.org/ftp/Specs/html-info/23228.htm`.

[ANL01] Tuomas Aura, Pekka Nikander, and Jussipekka Leiwo. Dos-resistant authentication with client puzzles. In *Revised Papers from the 8th International Workshop on Security Protocols*, pages 170–177, London, UK, 2001. Springer-Verlag.

[Ber05] Tom Berson. Skype security evaluation. Technical report, Anagram Laboratories, 2005.

[BLJ05] David A. Bryan, Bruce B. Lowekamp, and Cullen Jennings. Sosimple: A serverless, standards-based, p2p sip communication system. In *AAA-IDEA '05: Proceedings of the First International Workshop on Advanced Architectures and Algorithms for Internet Delivery and Applications*, pages 42–49, Washington, DC, USA, 2005. IEEE Computer Society.

[BM03] Colin Boyd and Anish Mathuria. *Protocols for Authentication and Key Establishment*. Springer, 1 edition, September 2003. Available from: `http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20\&path=ASIN/3540431071`.

[BM07] Ingmar Baumgart and Sebastian Mies. S/kademlia: A practicable approach towards secure key-based routing. In *ICPADS '07: Proceedings of the 13th International Conference on Parallel and Distributed Systems*, pages 1–8, Washington, DC, USA, 2007. IEEE Computer Society.

[Bry07] David A. Bryan. P2psip: On the road to a world without servers. *BUSINESS COMMUNICATIONS*, pages 40–44, April 2007.

[BS06] Salman Baset and Henning Schulzrinne. An analysis of the skype peer-to-peer internet telephony protocol. In *INFOCOM*. IEEE, 2006. Available from: `http://dblp.uni-trier.de/db/conf/infocom/infocom2006.html#BasetS06`.

[BS07] Andreas Binzenhöfer and Holger Schnabel. Improving the Performance and Robustness of Kademlia-based Overlay Networks. In *KIVS 2007*, Bern, Switzerland, February 2007.

[BSE09] Sheila Becker, Radu State, and Thomas Engel. Using game theory to configure P2P SIP. In *Proceedings of IPTComm '09, Atlanta, Georgia*, pages 1–9. ACM, 2009.

[Cam01] Gonzalo Camarillo. *SIP Demystified*. McGraw-Hill Professional, 2001. Foreword By-Rosenberg, Jonathan.

[CMM02] Russ Cox, Athicha Muthitacharoen, and Robert Morris. Serving DNS Using a Peer-to-Peer Lookup Service. In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 155–165, London, UK, 2002. Springer-Verlag.

[DA06] Zoran Despotovic and Karl Aberer. P2p reputation management: probabilistic estimation vs. social networks. *Comput. Netw.*, 50(4):485–500, 2006.

[DH00] Jochen Dinger and Hannes Hartenstein. Defending the sybil attack in p2p networks: Taxonomy, challenges, and a proposal for self-registration, April 200. Available from: http://dsn.tm.uni-karlsruhe.de/medien/publication-confs/dinger_dasp2p06_sybil.pdf.

[DLITT05] George Danezis, Chris Lesniewski-laas, Lin Tong, and Lin Tong. Sybil-resistant dht routing. In *In ESORICS*, page 305–318. Springer, Springer, 2005. Available from: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.65.3947.

[Dou02] John R. Douceur. The sybil attack. In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 251–260, London, UK, 2002. Springer-Verlag. Available from: http://portal.acm.org/citation.cfm?id=687813.

[DY81] D. Dolev and A. C. Yao. On the security of public key protocols. In *SFCS '81: Proceedings of the 22nd Annual Symposium on Foundations of Computer Science*, pages 350–357, Washington, DC, USA, 1981. IEEE Computer Society.

[FENH10] Ali Fessi, Nathan Evans, Heiko Niedermayer, and Ralph Holz. Pr2-P2PSIP: Privacy Preserving P2P Signaling for VoIP and IM. In *Principles, Systems and Applications of IP Telecommunications (IPTComm), Munich*, August 2010.

[FS00] Niels Ferguson and Bruce Schneier. A cryptographic evaluation of ipsec. Technical report, Counterpane Internet Security, Inc, 2000.

[GDJ06] Saikat Guha, Neil Daswani, and Ravi Jain. An experimental study of the skype peer-to-peer voip system. In *IPTPS'06: The 5th International Workshop on Peer-to-Peer Systems*, 2006.

[Hei06] VoIP-Störung bei United Internet [online]. July 2006. Available from: http://www.heise.de/newsticker/meldung/75382 [cited 2009-09-01].

[Hei10a] Angriff auf DNS-Server von InternetX [online]. January 2010. Available from: http://www.heise.de/newsticker/meldung/Angriff-auf-DNS-Server-von-InternetX-897712.html [cited 2010-06-04].

[Hei10b] Kundenanfragen überlasteten VoIP bei 1&1 [online]. November 2010. Available from: http://www.heise.de/newsticker/meldung/Kundenanfragen-ueberlasteten-VoIP-bei-1-1-Update-858101.html [cited 2010-06-04].

[Hei10c] VoIP-Ausfall bei 1&1 [online]. January 2010. Available from: http://www.heise.de/newsticker/meldung/VoIP-Ausfall-bei-1-1-Update-895875.html [cited 2010-06-04].

[HJP06] M. Handley, V. Jacobson, and C. Perkins. SDP: Session Description Protocol. RFC 4566 (Proposed Standard), July 2006. Available from: http://www.ietf.org/rfc/rfc4566.txt.

[Inca] CounterPath Inc. X-Lite. http://www.counterpath.com/x-lite.html. Last checked on Apr. 8th 2010, Last update on Mar. 2010.

[Incb] Drupal Inc. Ekiga Free Your Speech. http://ekiga.org/. Last checked on Apr. 8th 2010, Last update on Mar. 2010.

[JLR+10a] Cullen Jennings, Bruce Lowekamp, Eric Rescorla, Salman Baset, and Henning Schulzrinne. A SIP Usage for RELOAD. draft-ietf-p2psip-sip-04, Internet Draft, Work in Progress, March 2010. Available from: http://tools.ietf.org/html//draft-ietf-p2psip-sip-04.

[JLR+10b] Cullen Jennings, Bruce Lowekamp, Eric Rescorla, Salman Baset, and Henning Schulzrinne. REsource LOcation And Discovery (RELOAD). draft-ietf-p2psip-base-08, Internet Draft, Work in Progress, March 2010. Available from: http://tools.ietf.org/html/draft-ietf-p2psip-base.

[Joh03] Alan Johnston. *SIP: Understanding the Session Initiation Protocol, Second Edition*. Artech House, Inc., Norwood, MA, USA, 2003.

[KCTHK09] Hun Jeong Kang, Eric Chan-Tin, Nicholas Hopper, and Yongdae Kim. Why kad lookup fails. In *Peer-to-Peer Computing*, pages 121–130, 2009.

[LCR+07] Karthik Lakshminarayanan, Matthew Caesar, Murali Rangan, Tom Anderson, Scott Shenker, and Ion Stoica. Achieving convergence-free routing using failure-carrying packets. In *SIGCOMM '07: Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 241–252, New York, NY, USA, 2007. ACM.

[Lem02] Jonathan Lemon. Resisting syn flooding dos attacks with a syn cache. In *USENIX BSDCon'2002*, 2002.

[LKRG03] Dmitri Loguinov, Anuj Kumar, Vivek Rai, and Sai Ganesh. Graph-theoretic analysis of structured peer-to-peer systems: routing distances and fault resilience. In *SIGCOMM '03: Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 395–406, New York, NY, USA, 2003. ACM.

[LRW03] Nathaniel Leibowitz, Matei Ripeanu, and Adam Wierzbicki. Deconstructing the kazaa network. In *WIAPP '03: Proceedings of the The Third IEEE Workshop on Internet Applications*, page 112, Washington, DC, USA, 2003. IEEE Computer Society.

[Mas06] Dan Massey. A Comparative Study of the DNS Design with DHT-Based Alternatives. In *In the Proceedings of IEEE INFOCOM'06*, 2006.

[MGM05] Sergio Marti and Hector Garcia-Molina. Taxonomy of trust: Categorizing p2p reputation systems. Working Paper 2005-11, Stanford InfoLab, 2005. Available from: http://ilpubs.stanford.edu:8090/675/.

[MM02] Petar Maymounkov and David Mazieres. 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*, pages 53–65, 2002.

[New07] New York Times: Restarts Cited in Skype Failure [online]. August 2007. Available from: http://www.nytimes.com/2007/08/21/business/worldbusiness/21skype.html [cited 2009-09-01].

[Nic06] Saverio Niccolini. SPIT Prevention: State of the Art and Research Challenges. In *the 3rd VoIP Security Workshop*, Berlin, 2006.

[P2P] IETF P2PSIP working group charter web page. `http://www.ietf.org/dyn/wg/charter/p2psip-charter.html`. Last checked on Feb. 10th 2010.

[PPPW04] KyoungSoo Park, Vivek S. Pai, Larry Peterson, and Zhe Wang. Codns: improving dns performance and reliability via cooperative lookups. In *OSDI'04: Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation*, pages 14–14, Berkeley, CA, USA, 2004. USENIX Association.

[PW03] Birgit Pfitzmann and Michael Waidner. Federated identity-management protocols. In Bruce Christianson, Bruno Crispo, James A. Malcolm, and Michael Roe, editors, *Proceedings of 11th International Workshop on Security Protocols*, volume 3364/2005, pages 153–174. Springer-Verlag, LNCS, April 2003.

[Qut] QuteCom, Free VoIP Softphone. `http://www.qutecom.org/`. Last checked on Apr. 8th 2010, Last update on Mar. 2010.

[RD01a] Antony Rowstron and Peter Druschel. Pastry: Scalable, decentralized object location and routing for large-scale peer-to-peer systems. In *Lecture Notes in Computer Science*, pages 329–350, 2001.

[RD01b] Antony I. T. Rowstron and Peter Druschel. Storage Management and Caching in PAST, A Large-scale, Persistent Peer-to-peer Storage Utility. In *Symposium on Operating Systems Principles*, pages 188–201, 2001. Available from: `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.28.7055`.

[RGK+05] Sean Rhea, Brighten Godfrey, Brad Karp, John Kubiatowicz, Sylvia Ratnasamy, Scott Shenker, Ion Stoica, and Harlan Yu. Opendht: a public dht service and its uses. In *SIGCOMM '05: Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 73–84, New York, NY, USA, 2005. ACM.

[RGRK04] Sean Rhea, Dennis Geels, Timothy Roscoe, and John Kubiatowicz. Handling churn in a dht. In *ATEC '04: Proceedings of the annual conference on USENIX Annual Technical Conference*, pages 10–10, Berkeley, CA, USA, 2004. USENIX Association.

[Ros04] J. Rosenberg. A Presence Event Package for the Session Initiation Protocol (SIP). RFC 3856 (Proposed Standard), August 2004. Available from: `http://www.ietf.org/rfc/rfc3856.txt`.

[Ros10] J. Rosenberg. Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols. IETF RFC 5245 (Proposed Standard), April 2010. Available from: `http://www.ietf.org/rfc/rfc5245.txt`.

[RS04] Venugopalan Ramasubramanian and Emin Gün Sirer. The design and implementation of a next generation name service for the Internet. In *SIGCOMM '04: Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 331–342, New York, NY, USA, 2004. ACM.

[RSC+02] J. Rosenberg, H. Schulzrinne, G. Camarillo, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler. SIP: Session Initiation Protocol. RFC 3261, June 2002. Updated by RFCs 3265, 3853, 4320, 4916, 5393, 5621, 5626, 5630. Available from: `http://www.ietf.org/rfc/rfc3261.txt`.

[SCFJ03] H. Schulzrinne, S. Casner, R. Frederick, and V. Jacobson. RTP: A Transport Protocol for Real-Time Applications. RFC 3550 (Standard), July 2003. Updated by RFC 5506. Available from: `http://www.ietf.org/rfc/rfc3550.txt`.

[Sch95] Flora Rheta Schreiber. *Sybil*. Warner Books, 1995.

[Sch08] Henning Schulzrinne. Engineering Peer-to-Peer Systems. `http://www.p2p08.org/program/sessions/1-invited-talk-i/P2P08-keynote.pdf/at_download/file`, september 2008. Key note at the 8th IEEE International Conference on Peer-to-Peer Computing (P2P'08). Available from: `http://www.p2p08.org/program/sessions/1-invited-talk-i/P2P08-keynote.pdf/at_download/file`.

[See06a] Jan Seedorf. Security challenges for peer-to-peer sip. *IEEE Network*, 20(5):38–45, 2006. Available from: `http://dblp.uni-trier.de/db/journals/network/network20.html#Seedorf06`.

[See06b] Jan Seedorf. Using Cryptographically Generated SIP-URIs to Protect the Integrity of Content in P2P-SIP. In *the 3rd VoIP Security Workshop*, Berlin, 2006.

[See08] Jan Seedorf. Lawful interception in p2p-based voip systems. In *Proceedings of IPTComm 2008, Heidelberg, Germany, July 1-2, 2008*, pages 217–235, Berlin, Heidelberg, 2008. Springer-Verlag.

[SENB07] Moritz Steiner, Taoufik En Najjary, and Ernst W Biersack. A global view of KAD. In *IMC 2007, ACM SIGCOMM Internet Measurement Conference, October 23-26, 2007, San Diego, USA*, 10 2007.

[Sin06] Kundan Narendra Singh. *Reliable, Scalable and Interoperable Internet Telephony*. PhD thesis, Columbia University, New York, NY, 206.

[SIP] About SIP Express Router. `http://www.iptel.org/ser`. Last checked on Apr. 8th 2010; last updated Oct. 2007.

[SM02a] Emil Sit and Robert Morris. Security considerations for peer-to-peer distributed hash tables. In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 261–269, London, UK, 2002. Springer-Verlag.

[SM02b] Emil Sit and Robert Morris. Security considerations for peer-to-peer distributed hash tables. In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 261–269, London, UK, 2002. Springer-Verlag.

[SMK+01] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for Internet applications. In *Proceedings of ACM SIGCOMM 2001*, pages 149–160, New York, NY, USA, 2001.

[Sou] Sourceforge. KPhone. `http://sourceforge.net/projects/kphone/`. Last checked on Apr. 8th 2010.

[SS04] Kundan Singh and Henning Schulzrinne. Peer-to-peer internet telephony using sip. Technical report, Columbia University CUCS-044-04, 2004.

[TBC+05] Chunqiang Tang, Melissa J. Buco, Rong N. Chang, Sandhya Dwarkadas, Laura Z. Luan, Edward So, and Christopher Ward. Low traffic overlay networks with large routing tables. In *SIGMETRICS '05: Proceedings of the 2005 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, pages 14–25, New York, NY, USA, 2005. ACM.

[Tur10] S. Turner. The application/pkcs10 Media Type. IETF RFC 5967 (Informational), August 2010. Available from: `http://www.ietf.org/rfc/rfc5967.txt`.

[wel07] Am Abend waren die Telekom-Leitungen tot [online]. October 2007. Available from: `http://www.welt.de/wirtschaft/webwelt/article1313712/Am_Abend_waren_die_Telekom_Leitungen_tot.html` [cited 2010-06-02].

[WZH05] Honghao Wang, Yingwu Zhu, and Yiming Hu. An efficient and secure peer-to-peer overlay network. In *LCN '05: Proceedings of the The IEEE Conference on Local Computer Networks 30th Anniversary*, pages 764–771, Washington, DC, USA, 2005. IEEE Computer Society.

Towards Resilient Virtual Service Migration

Andreas Fischer, Yahya Al-Hazmi, and Hermann de Meer

Computer Networks and Computer Communications Lab

University of Passau, 94032 Passau, Germany

firstname.lastname@uni-passau.de

Abstract

Network resilience is an increasingly important topic for Future Internet research. Virtualization of services can help in increasing the resilience of a system through abstraction from hardware. This enables migration of services with a variety of migration strategies. However, resilience requirements of services can vary widely. There is, therefore, a need for managing service migration in a resilient way, picking the best strategy for service migration. This paper presents an architecture for resilient service migration, taking into account changing service requirements and different properties of migration strategies.

1. Introduction

With the global internet hosting more and more critical services, the resilience of that structure has gained increased attention. Our society increasingly depends on networked services. It is therefore important to find novel ways to strengthen and increase resilience of these services. We present here an architecture for service migration, focusing on the resilience of the migrated service. The services that are considered in this context are network services like DNS, HTTP, DHCP, Multimedia streaming, and similar services. Many of these services can become critical in the IT infrastructure of any company. It is thus important to not only discuss resilience of the underlying network, but to include the resilience of these services in order to assess the overall resilience of a given networked system. Resilience, in this context, means the ability of a system to quickly recover from any negative disturbance, avoiding a major loss of functionality. We use the D^2R^2+DR strategy presented in [7] as a model for achieving resilience. This strategy consists of two control loops (cf. Figure 1). The inner loop identifies defense, detection, remediation, and recovery as the relevant building blocks for short-term resilience. On top of that, there is an outer control loop, consisting of self-diagnosis and refinement of the applied strategies.

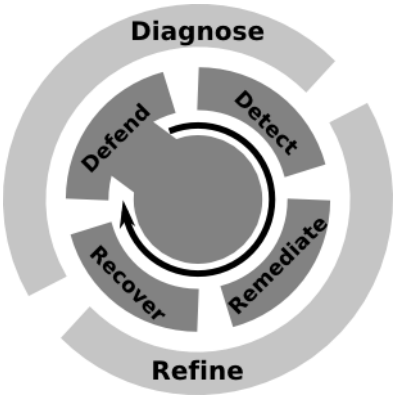


Figure 1: The D^2R^2+DR strategy

Applying these principles to network services, one way of increasing resilience of a network service is to virtualize it, thereby making it independent from the underlying hardware. A virtualized service can be migrated from one physical machine to another. This enables timely reaction to shortcomings of the underlying hardware.

Virtual services are envisioned to be manageable, since they are deployed in a virtual infrastructure which enables managing and optimizing services and resources usage transparently across the physical resources. As virtual services are not more than virtual machines, they can be created, terminated, cloned or migrated from physical machine to another. The migration process is one of the important management tasks in virtual infrastructure. It increases the flexibility of the whole infrastructure and also improves the infrastructure utilization by load balancing among the physical resources. A virtual service can be migrated for various reasons. The advantages of the migrating process become obvious in some use cases such as planned maintenance, removal of some physical resources (machines), and managing energy consumption.

Challenges leading to a service migration can stem both from hardware related problems as well as from software related ones. These challenges can be classified as follows:

1. **Hardware failure:** The physical machine a service is running on might fail for different reasons. Provided there is enough time to react, a migration of the service can help to keep up availability of the service. Examples of this category include:
    a. **True Failures:** This includes impending component failures, predictive notification of a natural catastrophe, etc.
    b. **Planned outages**: This includes scheduled maintenance, replacement of hardware components, etc.
2. **Service degradation:** Apart from outright failures, there may also be less critical shortcomings of the hardware – either long-term or short-term – which have to be remediated. These can be sub-classified as follows:
    a. **More hardware resources**: The service in question might, for example, need more CPU power in order to complete its assignments.
    b. **More network resources:** The service might need more bandwidth, or might have increased Quality of Service requirements.

c. **Energy Consumption Management:** The service might currently take up too much energy, requiring a migration to more energy-efficient physical machine.

A number of different migration strategies can be thought of. However, a suboptimal choice of the specific strategy may lead to undesirable effects, which may negatively influence the resilience of the service. Therefore, the way in which the migration process is executed is the key for keeping the service alive without any disruption. We present here an architecture that governs the migration process, enabling an optimal choice of migration strategies to remediate network challenges.

The rest of this paper is organized as follows: Section 2 discusses the options for resilient service migration, presenting different migration strategies and detailing the necessary steps for resilient service migration. In section 3, our architecture for resilient service migration is introduced. Section 4 presents related work. Finally, section 5 concludes this paper and gives an outlook on future work.

2. Migration of Services

Migration strategies can be classified according to different criteria. Looking at how the virtual machine is treated during the migration itself, one finds that migration can either happen live (i.e., without noticeable disruption of the service), with a paused virtual machine, or with a machine that has to be rebooted. A live migration guarantees minimal service downtime, but requires more bandwidth during migration. On the other hand, a cold migration, either with a paused machine or even a machine that was shutdown will introduce more downtime for the service, but is less complex to handle.

Looking at the availability of virtual machine images, migration strategies can either depend on common network storage for virtual machine images, pre-deployed boiler-plate images, or require image transfer during the migration process itself. Common network storage may not always be a viable solution, but reduces network traffic in certain scenarios. On the other hand, copying the entire virtual machine image during the migration process provides minimal complexity, but taxes the bandwidth quite noticeably. Pre-deployed boiler-plate images enable transmission of only the accumulated differences during migration, but require significant preparation before a challenge is even detected.

After the virtual machine image has been copied, service requests have to be rerouted in order to find the service at its new location. This involves some modification of the network. Such a modification can happen at all network layers, with different consequences. The lower the change happens in the network, the more transparent, but also inflexible, is the solution. Fixing the routing on the Link Layer can be transparent to all other layers, but is limited to link-local migrations. On the other hand, routing service connections through a Peer-to-Peer network allow migration to virtually any position in the network, since the Peer-to-Peer network abstracts from the actual network topology. However, this requires modification of the service and all applications accessing it.

Going through all of these possible migration strategies, it became obvious that it will be necessary to measure the cost of different combinations of techniques in order to decide which one should be used to counter a particular challenge.

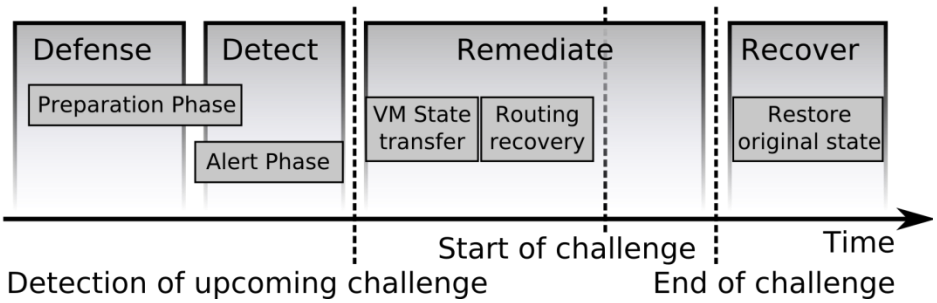


Figure 2: Phases of Virtual Machine migration

Several important steps have been identified as necessary when using virtual service migration as a resilience mechanism. They are depicted in Figure 2. The figure also illustrates the connection between migration and the D2R2 + DR strategy. As a defense measure there is first a preparation phase. This can be used, for example, to identify possible target hosts for migration, or to deploy boiler-plate virtual machine images that can be used to minimize the actual time to migrate.

Once a challenge is expected, challenge detection needs to start. This marks an “alert” phase, waiting for a challenge to happen. The actual migration of the virtual service is a remediation mechanism, and as such takes place after an adverse condition has been detected. Two integration points with other activities in ResumeNet have been identified at this position: Challenge Detection mechanisms are necessary in order to decide at which point to switch from Detection to Remediation, starting the migration process. It is necessary to keep in mind that, although migration time can be minimized by various measures, it can't be completely eliminated. Therefore, the challenge has to be indicated well before actual degradation of the system happens. Moreover, a Policy System has to provide input about acceptable target hosts and viable migration strategies.

Finally, after the challenge ends, the system has to recover, switching back to its original mode of operation. This includes a migration of the virtual service back to its original location.

2.1. Migration Strategies:

Different strategies for carrying out this migration of virtual services are possible. They differ from each other in the way they are executed which has an effect on the resilience of the service. They also have a different impact on the service with regard to the migration parameters they consider. Some of these strategies are presented below.

2.1.1. Cold migration:

In this strategy, the virtual machine is firstly suspended and then migrated to the target location. The migration itself encompasses transfer of the persistent state of the VM (i.e. the virtual hard drive), transfer of the volatile state of the VM (i.e. RAM contents and CPU state), and the redirection of network traffic. Once the state transfer is complete, it is resumed again in the new physical machine. During the migration time, the service is completely unavailable. This strategy is therefore suitable for services which do not have an impact on the users if they are not available for the short period of time (e.g. the migration downtime). This strategy is referred to as stop-and-copy.

An alternative cold migration strategy can be to completely power down the VM, transfer only the persistent state, and then power it up again at the new location. While this strategy has the obvious drawbacks of not being transparent for the VM and dropping all existing connections, it also has the advantage of being less complex. In particular, since all existing connections are dropped anyway, redirection of network traffic can be achieved trivially (e.g. via a combination of DHCP and DNS).

2.1.2. Live migration:

During live migration, a virtual service is migrated from one physical machine to another while maintaining availability of the service. Two types of live migration strategies are discussed here: pre-copy and post-copy. In pre-copy migration [2], the virtual machine's memory is first migrated to the target machine, while the service remains available in the source machine. If, during the migration, a memory page is changed on the source host, it is resent in the next round. Dirty pages are iteratively transferred in successive rounds until either a preset number of iterations is reached or the changeset has become reasonably small, allowing to interrupt the service for a very short time and transmit the remaining data together with the CPU state. The interruption necessary to copy the remaining memory pages constitutes the actual service downtime for this type of migration. The virtual machine is then restarted in the target machine.

An alternative strategy – the pre-copy strategy – focuses on reducing migration downtime as much as possible by minimizing the amount of virtual machine's state that is transferred during downtime. This is achieved well when the virtual service's workload is read intensive, but not for write intensive workload because pages that are repeatedly dirtied may have to be transmitted multiple times.

If the memory is transferred after transferring the CPU's state, this strategy referred to in [3] as post-copy migration. In this strategy, all processor state is firstly transmitted to the target machine, starts the virtual machine at the target, and then actively pushes memory pages from source to target. Once the virtual machine is resumed, its memory accesses leads to page faults, when such pages not yet pushed. These pages are requested from the source and transferred concurrently with pushed pages. To reduce the number of page faults, the occurrence of major faults can be predicted in advance using hints from the virtual machine's page access patterns, and hence adapt the page pushing sequence accordingly. Authors in [3] referred to this technique as "prepaging". Post-copy strategy ensures that each memory page is transferred at most once, and therefore avoiding the duplicate transmission overhead of pre-copy. Authors in [3] developed also so called dynamic self-ballooning mechanism to avoid transmitting free pages.

Live migration can easily be performed in datacenters, where rerouting of the network traffic is a minor issue. It gets more difficult (but is still possible) across several subnets, as seen in [4].

2.2. Resilience-related migration parameters

Several parameters related to virtual machine migration have to be considered for service resilience. First, there is the total amount of data that has to be transferred from the source host to the target host. Depending on the migration strategy, this may involve both, the persistent state of the virtual service, as represented by the associated virtual hard drive(s) and the volatile state, as represented by

memory contents and CPU status. The amount of data to be transferred also affects another important parameter indirectly: total time to complete migration. This encompasses the entire time span from the initiation of the migration procedure to the receiving of the last necessary data on the target host. In some cases (e.g. simple cold migration) this time is defined solely by the total amount of data to transfer, combined with the bandwidth that is available for the transfer. In other cases (e.g. live migration), things are more complicated, as more and more data has to be transferred until the migration is finally complete.

On the other hand, service downtime is obviously an important parameter for resilience considerations. For cold migration, this is identical with the total time to migrate. For live migration, the downtime is typically significantly shorter than the total time to migrate.

Finally, there is a binary parameter to consider – namely whether existing connections to the service can be kept or not. If the service in question is stateless (like a DNS server) this parameter can actually be ignored. On the other hand, if the service keeps state (like a Webmail server), users will be interrupted in their workflow, if existing connections are dropped.

3. Architecture for Resilient Service Migration

Network services have different properties, and hence different consideration should be taken into account while making service migration decisions. Therefore, we are looking for a management system which selects the more suitable strategy for migrating a particular service with specific properties and performance requirements. We argue that the selection of a suitable migration strategy leads to better service resilience.

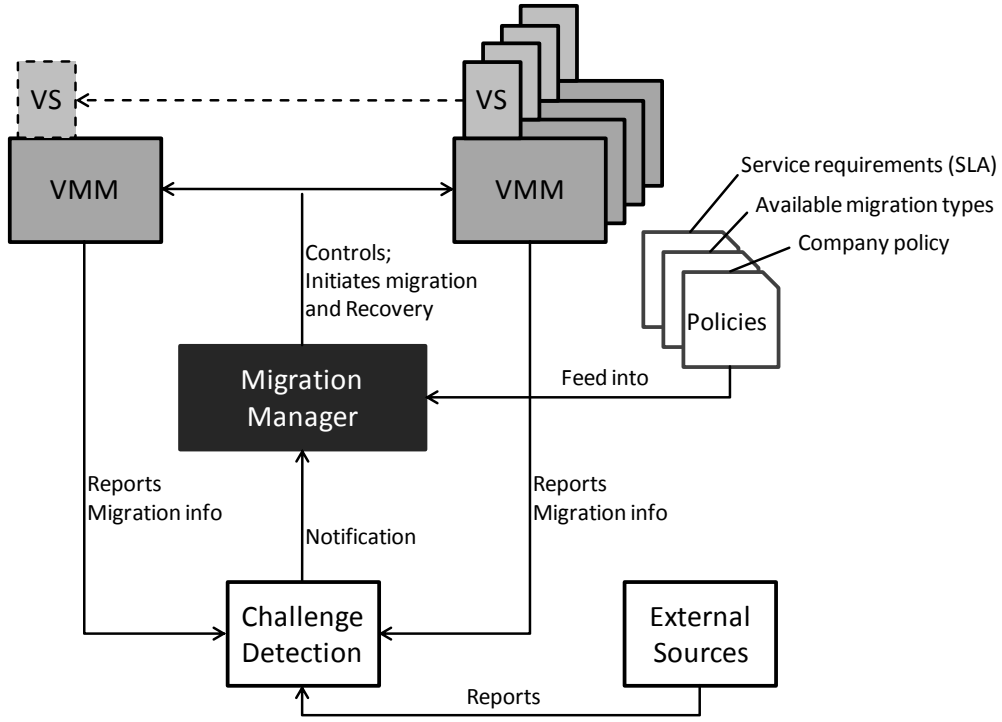


Figure 3 - Resilient virtual service migration architecture

We have designed an architecture that enables migrating network services, taking into account the resilience of the service. Figure 3 illustrates this architecture, which consists of several components. Central to the architecture is the concept of a Migration Manager. It controls and assigns hardware to Virtual Services (VSs) based on their current needs. In order to do so, it may trigger a migration of a Virtual Service from one physical machine to another. The Migration Manager is supported in its decision by a Challenge Detection component, which collects monitoring information from the physical machines (e.g. service degradation) and correlates it with information about challenges from external sources (information about HW failures, maintenance, managing energy consumption, etc). If a challenge is impending, it informs the Migration Manager, allowing the latter to initiate the appropriate migration in time. A policy system directs the actions of the Migration Manager, taking into account Service Level Agreements, available migration strategies, and other policies relevant for operation. When the challenge is over, the Migration Manager recovers the virtual service to its origin physical machine.

In this architecture, the Migration Manager is responsible for selecting the suitable migration strategy, but the first question arises that why not use one of the live migration strategies for all kind of services as long as this strategy is performed without disturbing the service availability. As discussed in the previous section, each strategy has different impact on migration parameters, and thus, services are differently affected. Therefore, one strategy could lead to a better service resilience than the other. The fundamental issue to be addressed here is how to measure the service resilience whether it is in an acceptable, impaired or unacceptable level as classified in [4]. To clarify this issue we need a new model which helps on specifying the service resilience levels based on the properties and requirements of the service.

To understand the concept, let's consider a video streaming server as an example for the virtual service. Firstly, we have to understand the service resilience levels. Most movies are shot at 24 frames per second (FPS). For human perception, about 16-18 FPS is sufficient to create the illusion of fluid motion. If the user receives the video with a rate of 24 FPS, we can therefore consider that the service is at an acceptable level. Between 16 FPS and 24 FPS, the service is at an impaired level, while receiving the video with less than 16 FPS means that the service is at an unacceptable level. Let's assume that this service requires a bandwidth of 10 Mbps, but the available bandwidth is only 5 Mbps. For this reason the streaming server has to be migrated. During the migration process, the available bandwidth (5 Mbps) will be shared by the running service and migrating the virtual machine's states. But using a live migration strategy means more time is needed for migration process. Consequently, service resilience could be negatively affected. Due to the small bandwidth, frames are received by the user in an unacceptable rate. Therefore, the user's buffer should be as large as possible to buffer the received frames and then play out the video in an acceptable level. If the buffer size is not large enough, the user is then under compulsion to play out the video in an unacceptable level (e.g. with a rate less than 16 FPS) to avoid packet loss. In this case, cold migration could be a better solution, since the available bandwidth (5Mbps) is only used for migrating the virtual machine while the service will suspend for a period of time. The larger is the video length the longer is this period.

4. Related Work

Service migration has been presented in several works [2, 5, 6 and 8], but they focus only on data centers. Marvin and et al. [5] designed a virtual machine management system called Usher which enables administrators to manage clusters of virtual machines, such as choosing the environment the virtual machine can be configured on, choose the policies under which they will be managed. The migration process is used to adapt to changes in load among active virtual machines or add and remove hardware. Asynchronous virtual machine replication is used by Remus [6] to gain high availability in the face of hardware failures. It provides a high degree of fault tolerance, so that a running system can transparently continue execution on an alternate physical machine with only seconds of downtime. Software is encapsulated in each virtual machine and asynchronously sends changed state to a backup machine at high frequencies. The authors in [8] present a cooperative, context-aware approach to data center migration across wide area networks to achieve high availability of data center services in the face of both planned and unanticipated outages. Server virtualization is used to enable the replication and migration of servers. To increase the availability of the services, the authors in [9] present a lightweight live migration mechanism to integrate whole-system migration and input replay efforts. Their goal is to reduce the overhead resulting from continuous live migration with checkpointing, while providing high service availability.

A migration process out of data centers has been presented by Yi Wang and et al. in [1]. They present a network-management primitive called VROOM for enabling live router migration. VROOM considers only a live migration strategy for migrating a particular service (routing service that is managed by routers). William and et al. [10] focus on studying the performance evaluation, focusing on the effects of live migration of virtual machines on the performance of modern Internet applications running in the virtual machines, such as multi-tier Web 2.0 applications.

In contrast to these works, we focus on the definition of a resilient service architecture for migrating different kinds of network services. This architecture is also able to select the most suitable strategy for migration of a particular service with specific properties while keeping the service resilient.

5. Conclusions

The investigation of possible migration strategies showed that there is no one-size-fits-all solution to each and every challenge. A migration strategy that proves to be viable against one type of challenge might prove suboptimal against other challenges. Therefore, it is necessary to weigh the specific challenge with the cost of different migration strategies in order to decide for the best remediation action. An architecture that enables autonomic selection between multiple options for virtual service migration, based on a number of resilience-related parameters of service migration, has been presented here.

One issue not discussed here is the interconnection of multiple network services. This may create further problems during service migration, as some services may actually depend on topological proximity. Future work will therefore include an investigation of the migration of entire (sub-)networks.

Moreover, we intend to show the practical application of our architecture, applying the demonstrated principles in existing network testbeds.

References

[1] Y. Wang, E. Keller, B. Biskeborn, J. van der Merwe and J. Rexford. Virtual Routers on the Move: Live Router Migration as a Network-Management Primitive. *In Proceedings SIGCOMM'08*, August 2008.

[2] C. Clark, K. Fraser, S. Hand, J. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield. Live migration of virtual machines. *In Proceedings NSDI'05*, May 2005.

[3] M. Hines, U. Deshpande, and K. Gopala., Post-Copy Live Migration of Virtual Machine. *In SIGOPS Operating Systems Review, Volume 43, Number 3, pages 14--26*, July 2009.

[4] Robert Bradford, Evangelos Kotsovinos, Anja Feldmann, Harald Schiöberg. Live Wide-Area Migration of Virtual Machines Including Local Persistent State. *In Proceedings VEE'07, pages 169 – 179*, June 2007.

[5] M. McNett, D. Gupta, A. Vahdat, and G. M. Voelker. Usher: An extensible framework for managing clusters of virtual machines. *In Proceedings USENIX LISA*, November 2007.

[6] B. Cully, G. Lefebvre, D. Meyer, M. Feeley, N. Hutchinson, and A. Warfield. Remus: High availability via asynchronous virtual machine replication. *In Proceedings NSDI*, April 2008.

[7] D. Hutchison and J.P.G. Sterbenz. ResiliNets: resilient and survivable networks. ERCIM News 77, April 2009

[8] K.K. Ramakrishnan, P. Shenoy , J. Van der Merwe. Live Data Center Migration acrossWANs: A Robust Cooperative Context Aware Approach. *In Proceedings INM'07, pages 262—267*, August 2007.

[9] B. Jiang, B. Ravindran, and C. Kim. Lightweight Live Migration for High Availability Cluster Service. *In Proceedings SSS 2010*, September 2010.

[10] W. Voorsluys, J. Broberg, S. Venugopal, and R. Buyya. Cost of Virtual Machine Live Migration in Clouds: A Performance Evaluation. *In CloudCom, Volume 5931, pages 254 – 265*, December 2009.