Bachelorarbeit

zur Erlangung des akademischen Grades

“Bachelor of Science in Engineering“

≪SIP und IPSec≫

ausgeführt von

Anton Paul Schmidbauer

Bahnhofplatz 8/9
3500 Krems an der Donau

Begutachter/in: DI (FH) Mag. Alexander Philipp Lintenhofer

Wien, den 27. Februar 2007

Ausgeführt an der Fachhochschule Technikum Wien

Studiengang: BICSS

FACHHOCHSCHULE
TECHNIKUM WIEN

Kurzfassung

Viele Internetbenutzer verwenden Voice over IP Dienste. Wenige machen sich aber über die Sicherheit der übertragenen Daten Gedanken.

Sichere Kommunikation soll Vertraulichkeit, Integrität und Authentizität der übertragenen Daten gewährleisten. Bei einer Voice over IP Anwendung sollen aber sowohl die Signalisierungsdaten als auch der Media Stream geschützt werden. Da beide auch unterschiedliche Wege im Netzwerk nehmen können, ist es leider nicht immer leicht erkennbar welche Teile der Kommunikation, wie abgesichert werden.

Diese Arbeit beschäftigt sich mit einer Möglichkeit, Voice over IP Daten zu schützen. Besonderes Augenmerk wurde dabei auf den Verlauf der Signalisierung und des Media Streams gelegt und deren Wege durch das Netzwerk.

Eidesstattliche Erklärung

“Ich erkläre hiermit an Eides Statt, dass ich die vorliegende Arbeit selbständig angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher weder in gleicher noch in ähnlicher Form einer anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.“

Inhaltsverzeichnis

1 Problem- und Aufgabenstellung

IPsec ist eine Sicherheitsarchitektur, die auf der Netzwerkschicht Sicherheitsdienste spezifiziert und somit den vertraulichen und authentifizierten Transport von IP- Datagrammen gewährleistet. Einerseits arbeitet es für die höheren Schichten transparent - andererseits sind hinsichtlich des Signalisierungs- und Mediaverkehrs Seiteneffekte zu berücksichtigen. Im Zuge dieser Arbeit soll der Verbund zweier LANs über ein Trägernetzëingerichtet und SIP-basierte Kommunikation über ein VPN getunnelt werden. Lösungsansätze für brauchbares Schlüsselmanagement sind genauso zu diskutieren wie sich ergebende sicherheitsspezifische Probleme bzw. Einschränkungen der Dienstegüte (Quality of Service).

2 Einleitung

Das Session Initiation Protokoll (SIP) ist ein ASCII Protokoll. Es kann damit von jedem mit geeigneter Ausrüstung kompromitiert werden. Besonders Internet Service Providern ist dies ein Leichtes. Mit der Internet Sicherheitsarchitektur IPSec bietet sich aber eine Möglichkeit, Voice over IP Dienste kryptographisch zu sichern. Anhand eines einfachen Beispiels soll untersucht werden, wie IPSec hier angewandt werden kann.

Gegeben ist ein Unternehmen mit 2 Standorten. Beide Standorte haben ein internes Netzwerk. Diese beiden Netzwerke sollen mit Hilfe von IPSec durch ein öffentliches Netzwerk verbunden werden. Innerbetriebliche Telefongespräche sollen über diese Verbindung übertragen werden, allerdings soll keine Möglichkeit bestehen, diese Gespräche abzuhören. Des Weiteren soll es beiden Standorten auch ermöglicht werden Teilnehmer im öffentlichen Netzwerk zu erreichen.

Zuerst wurde eine Voice over IP Verbindung zwischen den beiden privaten Netzwerken aufgebaut. Dabei wurde genau untersucht, welche Wege die Signalisierung und der Mediastream im Netzwerk nehmen. Danach folgte ein Verbindungsaufbau eines Clients im privaten Netz zu einem Verbindungspartner im öffentlichen Netz. Wieder wurde der Verlauf der Signalisierung und des Mediastreams genauestens analysiert. Abschließend wurde noch untersucht, wie sich Signalisierung und Media Stream bei einer Verbindung von einem Client im öffentlichen Netz zu einem Client im privaten Netzwerk verhalten.

Im Grundlagenteil werden die wesentliche Teile der IPSec Architektur erklärt. Danach folgt im praktischen Teil eine Beschreibung der Teststellung und der durchgeführten Tests. Die Ergebnisse der Tests folgen der Beschreibung.

3 Grundlagen

Die folgenden Abschnitte geben eine kurze Einführung in das Session Initiation Protokoll (SIP). Weiters folgt eine detaillierte Beschreibung der IP Sicherheitsarchitektur (IPSec).

3.1 Session Initiation Protokoll

SIP ist ein Protokoll, das dem Aufbau und der Verwaltung von Multimedia Session im Internet dient. Über SIP können z.B. Internet-Telefonverbindungen eingerichtet werden. SIP ermöglicht es, einen oder mehrere Sessionpartner zu einer Session einzuladen, wobei der Begriff "Session" jegliche Multimediaverbindung beinhaltet und nicht auf den Begiff "Telefonie" eingeschränkt ist.

SIP übernimmt dabei die Aufgaben des Auffindens, das Feststellen der Verfügbarkeit der Sessionpartner und das Ausverhandeln der Verbindungsparameter der Session.

SIP wird in RFC3261 [RSC+02] beschrieben.

3.1.1 SIP Komponenten

Das SIP Protokoll beschreibt folgende Komponenten, die beim Aufbau einer Session eine wesentliche Rolle spielen

- User Agent (UA): Dieser stellt sich entweder als User Agent Client (UAC), der eine SIP Verbindung initiiert, oder als User Agent Server (UAS), der auf den initialen Request antwortet, dar.
- Outbound Proxy: Der Outbound Proxy nimmt einen Request vom UAC entgegen und sucht via DNS Abfragen (SRV und NAPTR Records siehe RFC3263 [RS02]) den zuständigen Inbound Proxy für diesen Request.
- Inbound Proxy: Der Inbound Proxy nimmt einen Request vom Outbound Proxy entgegen und sucht in der Location Database (siehe weiter unten) nach dem zugehörigen UAS.
- Domain Name Service (DNS): Nach Erhalt eines Request vom UAC sucht der Outbound Proxy nach dem zugehörigen Inbound Proxy im DNS
- Location Database: SIP unterscheidet zwischen einer User URI, dem Address of Record (AoR) URI, der Contact Address. Der AoR ist die logische Adresse eines Teilnehmers. Dieser Adresse ist noch keine IP Adresse zugeordnet. Erst durch die Contact Adresse, die die physiche Adresse (IP Adresse) enthält, erreicht man ein Verbindung zwischen logischer und physischer Adresse. Der UAS registriert sich daher in der Location Database mit der Contact Adresse und dem AoR. Der Inbound Proxy kann die Lokation eines UAS mit Hilfe der Registrierungseinträge feststellen.

3.1.2 SIP Verbindungsaufbau

Da sich diese Arbeit hauptsächlich mit IPSec und dessen Zusammenhang mit dem Aufbau einer Verbindung via SIP beschäftigt, folgt eine kurze Beschreibung der Initialisierung einer SIP-Session. Das folgende Beispiel in Abbildung 1 ist RFC3261 [RSC+02] entnommen.

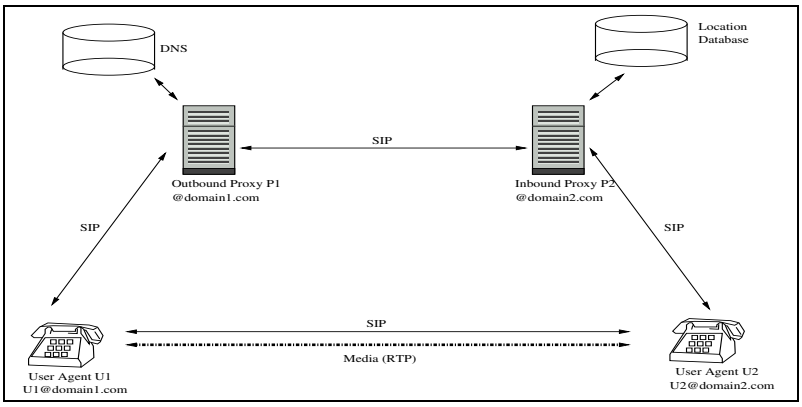

Abb. 1: SIP Trapezoid

User Agent U1 möchte eine Verbindung zu User Agent U2 aufbauen. Zuerst wird ein `INVITE sip:u2@domain2.com` Request an den Outbound Proxy P1 gesendet. Der Outbound Proxy P1 sucht via DNS den zuständigen Inbound Proxy P2 für `domain2.com` und verschickt die Nachricht an diesen. Da der Inbound Proxy P2 für die Domain `domain2.com` zuständig ist, sucht dieser in der Location Database nach der Kontaktadresse des User Agent U2 und leitet den Request weiter. U2 antwortet mit einem `OK` und schickt diese Nachricht wieder über P2 und P1 an den User Agent U1 zurück. Wenn der `OK` Request beim User Agent U1 eingelangt ist, beginnt die Session zwischen U1 und U2. Jede weitere Kommunikation zwischen den beiden User Agents verläuft direkt.

Das besondere an SIP ist, dass Signalisierung und Media Stream nicht dem gleichen Pfad folgen. SIP wird in RFC3261 [RSC+02] spezifiziert.

3.2 IP Security Protokoll (IPSec)

Das IP Security Protokoll (IPSec) wird in RFC4301 [KS05] beschrieben. IPSec soll Zugriffkontrolle, Datenintegrität, Authentifizierung der Datenquelle sowie Schutz vor sogenannten Replay-Attacken (Wiedereinspielen von bereits versendeten Paketen) und Vertraulichkeit auf der Netzwerkschicht (OSI Layer 3) gewährleisten. Die IPSec Architektur besteht aus den Protokollen Encapsulated Security Payload (ESP) und Authentication Header (AH). Zum automatischen Schlüsselaustausch zwischen zwei IPSec Endpunkten werden die Protokolle Internet Security Association and Key Managment Protokoll (ISAKMP) und Internet Key Exchange Protokoll (IKE) spezifiziert.

Die IPSec Architektur unterscheidet zwischen Transport- und Tunnel Mode. Der Transport Mode definiert eine Host zu Host Verbindung, die mit IPSec abgesichert wird, der Tunnel

Mode definiert eine Host zu Netwerk oder Netzwerk zu Netzwerk Verbindung, die mit IPSec geschützt wird.

3.2.1 Security Policy

Um entscheiden zu können, welche Sicherheitsprotokolle (ESP oder AH im Transport oder Tunnel Mode) für eine bestimmte Verbindung zwischen Sender und Empfänger angewandt werden sollen, wird eine Security Policy festgelegt.

Listing 1: Security Policy

```
1  0.0.0.0/0[any] 192.168.170.32/28[any] any
           in ipsec
3          esp/tunnel/192.168.170.33-192.168.170.36/require
           created: Dec 20 15:09:29 2006  lastused: Dec 20 15:12:38 2006
5          lifetime: 0(s) validtime: 0(s)
           spid=16393 seq=3 pid=83738
7          refcnt=3

9  192.168.170.32/28[any] 0.0.0.0/0[any] any
           out ipsec
11         esp/tunnel/192.168.170.36-192.168.170.33/require
           created: Dec 20 15:09:29 2006  lastused: Dec 20 15:12:38 2006
13         lifetime: 0(s) validtime: 0(s)
           spid=16392 seq=0 pid=83738
15         refcnt=3
```

Listing 1 stellt zwei Security Policies dar. Die erste Policy beschreibt einen IPSec Tunnel mit den Sourceadressen (`0.0.0.0/0`) und den Zieladressen `192.168.170.32/28` (Zeile 1). Diese Policy betrifft eingehende IP-Pakete (Zeile 2), die mit ESP im Tunnel Mode zwischen den IPSec Servern `192.168.170.33` und `192.168.170.36` abzusichern sind (Zeile 3).

Die zweite Policy beschreibt den oben beschriebenen Tunnel in der Gegenrichtung und wird beim Versenden von IP Paketen verwendet. Man erkennt, dass SP's immer unidirektional selektieren. Security Policies werden in der Security Policy Database (SPD) gespeichert.

3.2.2 Security Associations (SA)

Wie in RFC4301 [KS05] beschrieben, sind Security Associations (SA) ein wesentlicher Bestandteil von IPSec. Eine SA definiert unidirektional das Protokoll (ESP oder AH) und die Schlüssel zum Ver- und Entschlüsseln von Paketen, die zwischen Sender und Empfänger zur Anwendung kommen. Für eine bidirektionale Verbindung sind zwei SA's notwendig, eine SA vom Sender zum Empfänger und eine in Gegenrichtung. Sollen sowohl ESP als auch AH angewandt werden, sind für beide Protokolle unterschiedliche SA's auszuverhandeln.

Eine Security Association wird durch folgende Parameter definiert:

- Security Parameter Index (SPI) : Ein 32bit Wert Index, der die Zuordnung von IP-Paketen zur zugehörigen SA ermöglicht.
- AH Authentifzierungsalgorithmus und der zugehörige Schlüssel (wenn AH verwendet wird).
- ESP Authentifizierungsalgorithmus und der zugehörige Schlüssel (wenn ESP Authentifizierung verwendet wird).

- ESP Integritätsalgorithmus und der zugehörige Schlüssel (wenn ESP zur Sicherung der Integrität von IP Paketen verwendet wird).
- ESP Authentifizierungs- und Integritätsalgorithmus und zugehörige Schlüssel (wenn ESP zur Authentifizierung und Integritätssicherung von IP Paketen verwendet wird).
- Lebenszeit der SA: Nach Ablauf eines Timers muss die SA und der zugehörige SPI neu ausverhandelt werden.

Die Security Associations werden in der Security Association Database abgelegt.

Listing 2: Security Association

```
1  192.168.170.36 192.168.170.33
       esp mode=tunnel spi=120167668(0x07299cf4) reqid=0(0x00000000)
3      E: 3des-cbc  2280a815 2cffaaea 0e112358 3f6f9333 c175c3f9 a17f6c27
       A: hmac-sha1  500e8b4d e170aeca c3251814 22770f78 e328ef44
5      seq=0x0000006b replay=4 flags=0x00000000 state=mature
       created: Dec 20 15:09:56 2006   current: Dec 20 15:13:47 2006
7      diff: 231(s)    hard: 300(s)    soft: 240(s)
       last: Dec 20 15:13:46 2006      hard: 0(s)      soft: 0(s)
9      current: 22408(bytes)   hard: 0(bytes)  soft: 0(bytes)
       allocated: 107  hard: 0 soft: 0
11     sadb_seq=1 pid=83995 refcnt=2
```

In Listing 2 ist der Inhalt einer SAD dargestellt. Die abgespeicherte SA beschreibt einen Tunnel (`mode=tunnel`) der mit ESP abgesichert wird (Zeile 2). Weiters ist in Zeile 3 und 4 der Verschlüsselungsalgorithmus (E:) und der Authentifizierungsalgorithmus (A:) angeführt. In Zeile 5 ist die aktuelle Sequence Number ersichtlich `seq=0x0000006b`, die zum Schutz vor Replay Attacken dient.

3.2.3 Zusammenspiel SA, SP

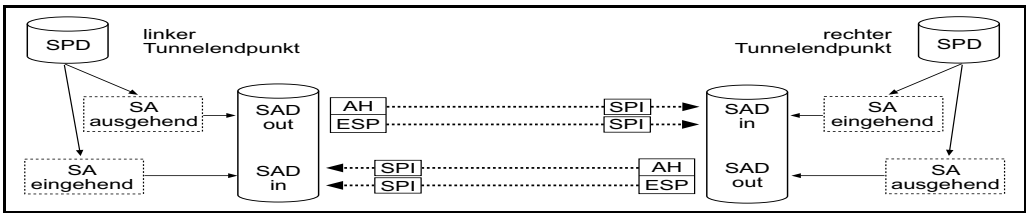

Abb. 2: Zusammenspiel SA, SAD, SDP [Lin06]

Durch eine Security Policy wird festgelegt, welche Sicherheitsprotokolle mit welchen Eigenschaften zwischen IPSec Servern zur Anwendung kommen. Diese Policy wird in der Security Policy Database abgelegt. Aus der Policy wird die Security Association abgeleitet. Eine SA beschreibt eine spezifische IPSec Verbindung zwischen zwei IPSec Endpunkten inklusive der verwendeten Algorithmen und deren Schlüssel. Gespeichert werden SA's in der Security Association Database. Der Security Parameter Index dient als Primärschlüssel in der SAD, um eingehende Pakete einer SA zuordnen zu können.

3.2.4 Encapsulated Security Payload (ESP)

Das ESP Protokoll wird in RFC4303 [Ken05b] beschrieben und stellt die Sicherheitsdienste Vertraulichkeit, Authentifizierung des Senders, Datenintegrität, Schutz vor Replay Attacken und Vertraulichkeit des Verbindungsablaufes (nur im Tunnel Mode,siehe 3.2.6) und wenn genügend Daten zwischen IPSec Verbindungspartnern ausgetauscht werden) zur Verfügung.

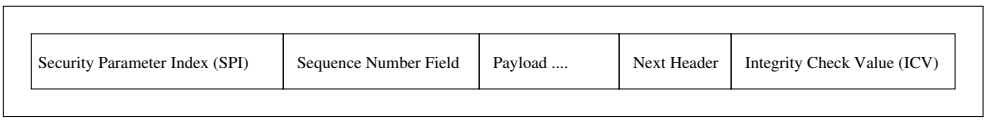

Abb. 3: ESP Paket

Abbildung 3 veranschaulicht den Aufbau eines ESP-Paktes:

- Der Security Parameter Index (SPI) dient dem Empfänger, um das Paket einer SA zuordnen zu können.
- Die Sequence Number dient zum Schutz vor Replay Attacken und entspricht einer per-SA Sequence Number. Bei jedem ESP-Paket, das verschickt wird, muss die Sequence Number erhöht werden.
- Der Type der Payload wird durch das Next Header Feld bestimmt. Wird ein Verschlüsselungsalgorithmus verwendet, der einen Initialization Vector (IV) benötigt, findet sich dieser IV ebenfalls in der Payload wieder.
- Das Next Header Feld identifiziert den Typ der Payload nach dem ESP Header. Dies kann z.B. IPv4 (Typ 4) oder TCP (Typ 6) sein.
- Der Integrity Check Value (ICV) ist die Prüfsumme des Paketes. Welcher Algorithmus zur Anwendung kommt, wird durch die SA festgelegt. Algorithmen können z.B. SHA256, AES, SHA1 oder MD5 sein.

3.2.5 Authentication Header (AH)

Das Authentication Header Protokoll wird in RFC4302 [Ken05a] beschrieben und soll Datenintegrität, Authentifizierung des Senders und Schutz vor Replay Attacken gewährleisten.

Da ESP im Tunnel Mode die gleichen Dienste bietet, wird in der Praxis meist auch nur ESP verwendet. Beim Aufbau der Teststellung für diese Bachelorarbeit kommt ebenfalls nur ESP zum Einsatz. Es wird daher hier nur der Vollständigkeit halber auf AH eingegangen.

AH versucht eine möglichst große Anzahl von Feldern des IP-Headers zu schützen. Da einige Felder aber durch Router verändert werden (z.b. Time-to-Live (TTL)), sind diese vom Schutz durch AH ausgenommen.

In Abbildung 4 ist der Aufbau eines AH Paketes dargestellt. Die Felder des Paketes haben die gleiche Bedeutung wie die eines ESP Paketes (Vergleiche 3.2.4).

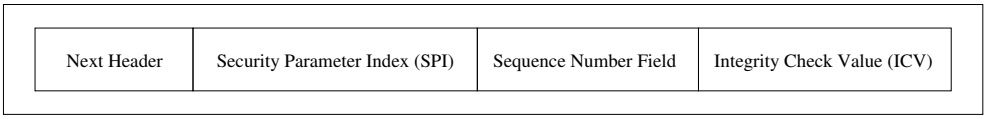

Abb. 4: AH Paket

3.2.6 Transport / Tunnel Mode

IPSec kann zum Schutz von Host-zu-Host (Transport-Mode), Host-zu-Netzwerk (Tunnel-Mode) und Netzwerk-zu-Netzwerk (Tunnel-Mode) Verbindungen verwendet werden.

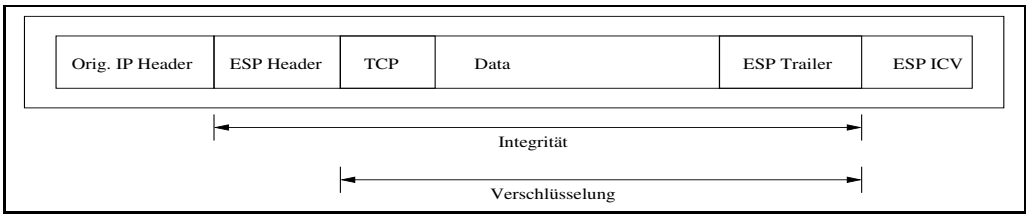

Abb. 5: ESP Paket im Transport Mode

Abbildung 5 stellt ein IPv4 Paket, das im Transport Modus übertragen wird, dar. Nach dem originalen IP-Header, der unverändert bleibt, folgt der ESP Header wie in 3.2.4 beschrieben. In diesem Beispiel hätte das ESP Next Header Feld den Wert 6, da auf den ESP Header ein TCP Paket folgt. Auf das TCP Paket und die Nutzdaten (Data) folgt noch der ESP Trailer und der ESP IV. Die Integrität des Paketes wird für die Felder ESP Header, TCP, Data und ESP Trailer gewährleistet. Verschlüsselt sind nur die Felder TCP, Data und ESP Trailer. Dies ist auch der wesentliche Unterschied zum Authentication Header Protokoll, das keine Verschlüsselung bietet, aber die Integrität auch für Teile des originalen IP-Headers gewährleistet.

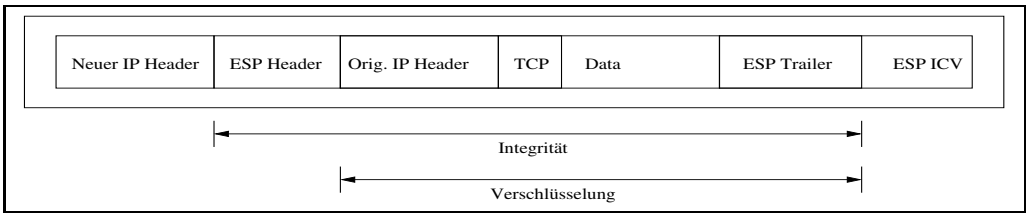

Abb. 6: ESP Paket im Tunnel Mode

Wie in Abbildung 6 dargestellt, ändert sich der Aufbau des Paketes im Tunnel Modus. Der originale IP-Header wird in die Payload verschoben. Im Unterschied zum Transport Modus werden sowohl Integrität, als auch Vertraulichkeit für den originalen IP-Header gewährleistet. ESP bietet im Tunnel Mode somit die gleichen Dienste wie AH.

3.3 Internet Security Association and Key Managment Protokoll / (ISAKMP) / Internet Key Exchange Protokoll (IKE)

Die Protokolle ISAKMP und IKE werden verwendet, um IPSec SA's in einer abgesicherten Weise auszuverhandeln. SA's werden von IPSec Verbindungspartner benötigt, um IP Pakete mit zuvor via IKE ausverhandelten Algorithmen zu ver- und entschlüsseln (siehe 3.2.2).

Das ISAKMP Protokoll ist in RFC2408 [MSST98] spezifiziert. Es beschreibt eine Möglichkeit, SA's zu verwalten (ausverhandeln, hinzufügen, ändern und löschen). ISAKMP ist kein spezielles Protokoll zum Austausch von Schlüsseln, es stellt lediglich ein Framework zur Verfügung. Dies bedeutet, dass ISAKMP von speziellen Verschlüsselungsalgorithmen, Algorithmen zum Erzeugen von kryptografischen Schlüsseln und Security Mechanismen losgelöst ist. ISAKMP kann zum Ausverhandeln von SA's für unterschiedliche Dienste verwendet werden (z.b. OSPF, IPSec usw.).

ISAKMP soll unter anderem vor Denial of Service (DoS) , Übernahme von Verbindungen (Hijacking) und Man in the Middle Attacken schützen.

Das Internet Key Exchange Protokoll Version 1 (IKEv1) wird in RFC 2409 [HC98] beschrieben. Diese RFC wurde durch den RFC4306 (IKEv2) [Kau05] abgelöst. Da OpenBSD aber derzeit nur IKEv1 implementiert, wird hier nur diese Version erläutert. IKE definiert ein Protokoll zum sicheren Erzeugen von kryptografischen Schlüsseln für SA's. Dies wird durch die Kombination der beiden Protokolle OAKLEY [Orm98] und SKEME [Kra96] erreicht.

Ebenso wie ISAKMP ist IKE ein generisches Protokoll, das zum Generieren und Ausverhandeln von Sessionschlüsseln für alle möglichen Anwendungsfälle gedacht ist (z.B. OSPF, SNMPv3, IPSec usw.). Für welche Anwendung IKE die Schlüsseln ausverhandeln soll, wird in einem eigenen Dokument festgelegt, einer sogenannten Domain of Interpretation (DOI) . Für IPSec legt diese Domain of Interpretation der RFC2407 [Pip98] fest.

Der Verbindungsaufbau zwischen 2 Partnern via ISAKMP wird in zwei Phasen unterteilt.

Phase 1 legt die sogenannte ISAKMP Security Association fest. Diese muss pro Verbindung zwischen zwei ISAKMP-Servern nur einmal ausverhandelt werden. Alle weiteren SA's werden von dieser abgeleitet.

Als Attribute werden der Verschlüsselungsalgorithmus festgelegt, der Hash Algorithmus, die Authentifizierungsmethode und die Diffie-Hellman Gruppe (Festlegung der öffentlichen Werte g und p. g ist in den meisten Fällen die Dezimalzahl 2, Gruppe 2 definiert eine Primzahl p mit der Länge 1024 Bit, Gruppe 5 verwendet 1536 Bit).

Da die Phase 1 dem Aufbau einer abgesicherten Verbindung dient, wird IKE zum Generieren und dem Austausch von Schlüsseln benötigt. Es wird hier zwischen zwei Varianten unterschieden:

- Im Main Mode werden zwei Nachrichten für das Ausverhandeln der Policy, zwei Nachrichten zum Austausch der öffentlichen Deffie-Hellman Werte und zwei Nachrichten um den Deffie-Hellman Austausch zu authentifizieren versandt.
- Im Aggressive Mode werden bereits in den ersten beiden Nachrichten die Policy, die öffentlichen Deffie-Hellman Werte und die Identitäten ausgetauscht. Weiters authenti-

fiziert die zweite Nachricht den Client der Verbindung. Eine dritte Nachricht authentifiziert den Initiator des Austauschs und dient dem Client als Beweis, dass die Nachricht auch wirklich von diesem Initiator gesendet wurde.

Nachteil des Agressive Modes ist, dass keine Prüfung der Identität erfolgen kann. Diese werden vor dem Ausverhandeln eines gemeinsamen Schlüssels ausgetauscht und können daher nicht verschlüsselt werden.

Phase 2 dient dem Ausverhandeln der SA's für ein bestimmtes Service (in diesem Fall IPSec). Da sich diese Phase der bereits durch die ISAKMP Security Association gesicherten Verbindung bedient, wird dieser Modus "Quick Mode" genannt. Dieser bedient sich der bereits durch die ISAKMP Security Association gesicherten Verbindung und kann daher nur dann zum Einsatz kommen, wenn bereits eine ISAKMP SA besteht. Auf eine Authentifizierung kann Dank der vorhergehnden Phase 1 verzichtet werden, Quick Mode dient rein dem Austausch von Sessionschlüsseln.

IKE bietet als Authentifizierungsmöglichkeiten entweder Public Key Verfahren (RSA, Ditital Signature Standard (DSS)) oder einen zuvor definierten und allen Verbindungspartnern bekannten Schlüssel (Pre-Shared Key).

4 Praktischer Teil

Es wurde eine Teststellung aufgebaut und verschiedene Test durchgeführt. Zwei LANs mit RFC1918 [RMK+96] IP Adressen wurden über ein öffentliches Netzwerk miteinander verbunden. Diese Verbindung wurde mit IPSec im ESP Tunnel Modus gesichert. Untersucht wurde, welche Wege die Signalisierung und der Mediastream im Netzwerk nehmen.

Zuerst folgt eine Beschreibung des Testsetups und der verwendeten Software, danach die getesteten Szenarien und die Ergebnisse der durchgeführten Tests.

4.1 Testsetup

In diesem Teil werden die bei den Tests verwendeten Komponenten und deren Konfiguration kurz beschrieben.

4.1.1 OpenBSD

OpenBSD (`http://www.openbsd.org`, Stand 27. Februar 2007) ist ein auf der Berkeley Software Distribution basierendes frei verfügbares UNIX. Bei der Entwicklung wurde vor allem auf die Sicherheit des Systems geachtet. Daher eignet sich OpenBSD besonders als IPSec Gateway. Für diese Arbeit kam Version 4.0 zum Einsatz. OpenBSD implementiert den IPSec Stack im Kernelspace (Ring 0) des Betriebssystems. Zur Konfiguration und Administration des IPSec Stacks dient das Userspace Tool `ipsecctl(8)` und der Daemon `isakmpd`. Dieser Daemon implementiert die Protokolle ISAKMP und IKE (siehe 3.3). `ipsecctl` bedient sich der Konfigurationsdatei `/etc/ipsec.conf`. Dort werden IPSec Verbindungen konfiguriert. `isakmpd` wird ebenfalls durch die Datei `/etc/ipsec.conf` konfiguiert. Zusätzlich befinden sich aber noch weitere Konfigurationselemente in der Datei `/etc/isakmpd/isakmpd.policy`.

4.1.1.1 Konfiguration OpenBSD

Es wurde ein Tunnel zwischen zwei OpenBSD Gateways konfiguriert. Abbildung 7 zeigt die Netzwerkkonfiguration und die verwendeten IP-Adressen.

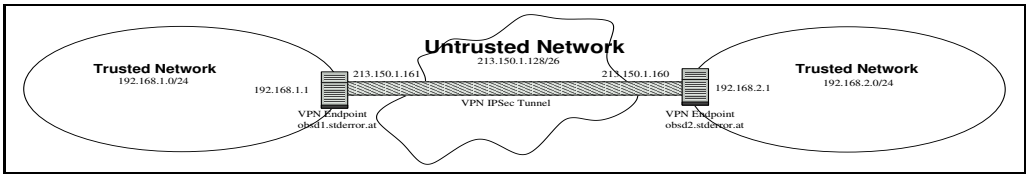

Abb. 7: Netzwerkkonfiguration OpenBSD

Die beiden OpenBSD Server haben die öffentlichen IP-Adressen `213.150.1.161` (obsd1.stderror.at) und `213.150.1.160` (obsd2.stderror.at). Diese beiden Adressen dienen

der Verbindung der beiden privaten Netzwerke `192.168.1.0/24` bzw. `192.168.2.0/24`.

Auf Server OBSD1 wurde folgender Policy in der Datei `/etc/ipsec.conf` definiert:

Listing 3: /etc/ipsec.conf Server OBSD1

```
1 ike active esp from 192.168.1.0/24 to 192.168.2.0/24 peer obsd2.stderror.at
```

Es sollen alle IP-Pakete, die zwischen den beiden Netzen `192.168.1.0/24` und `192.168.2.0/24` ausgetauscht werden, mit ESP im Tunnel Modus abgesichert werden. Die Verbindungsschlüssel werden automatisch mit IKE erzeugt. Als Verbindungspartner wird der Server `obsd2.stderror.at` definiert. Der Parameter `active` bewirkt, dass der ISAKMP Daemon, der die Protokolle ISAKMP und IKE (siehe 3.3) implementiert, sofort mit dem Ausverhandeln der SA's beginnt und nicht erst bei der ersten Anfrage des Verbindungspartners. Zu diesem Zweck müss der UDP Port 500 auf beiden IPSec Endpunkten erreichbar sein. Sollten die Server mit einer Firewall geschützt werden, muss darauf geachtet werden,dass eben dieser Port erreichbar ist.

OBSD2 definiert den gleichen Tunnel in die Gegenrichtung:

Listing 4: /etc/ipsec.conf Server OBSD2

```
1 ike active esp from 192.168.2.0/24 to 192.168.1.0/24 peer obsd1.stderror.at
```

Wie man leicht erkennen kann, wurden die IP-Adressen und der Name des Verbindungspartners ausgetauscht. Die Pakete werden in diesem Fall von dem Netzwerk `192.168.2.0/24` versandt und als Gegenstelle dient der Server `obsd1.stderror.at`.

Des Weiteren muss noch der Daemon `isakmpd` konfiguriert und gestartet werden. Die Konfiguration besteht aus der schon oben erwähnten Datei `/etc/ipsec.conf` und einer Datei `/etc/isakmpd/isakmpd.policy`. `isakmpd.policy` bestimmt welche Parameter die spezifischen Security Associations (vergleiche 3.2.2) enthalten müssen.

Listing 5: /etc/isakmpd/iskmpd.policy

```
1 Authorizer: "POLICY"
  Conditions: app_domain == "IPsec policy" &&
3             esp_present == "yes" &&
              esp_enc_alg != "null" -> "true";
```

Es sind nur IPSec SA's zu verhandeln, die ESP als Protokoll beinhalten und deren ESP Verschlüsselungsalgorithmus ungleich "null" ist, die also einen Verschlüsselungsalgorithmus verwenden (z.b. Advanced Encrypten Standard (AES)). `ISAKMPD` kann entweder manuell durch das Kommando `/sbin/isakmpd` oder automatisch beim Betriebssystemstart durch den Eintrag `isakmpd_flags = ''` in der Datei `/etc/rc.conf` gestartet werden.

Mit dem Userspace-Kommando `ipsecctl -f /etc/ipsec.conf` wird die Policy in die im Kernel befindliche Security Police Database geladen. `ipsectl -s flow` zeigt die definierten Policies an:

Listing 6: SPD

```
flow esp in from 192.168.2.0/24 to 192.168.1.0/24 peer 213.150.1.160
2    srcid 213.150.1.161/32 dstid 213.150.1.160/32 type use

4 flow esp out from 192.168.1.0/24 to 192.168.2.0/24 peer 213.150.1.160
     srcid 213.150.1.161/32 dstid 213.150.1.160/32 type require
```

Listing 6 zeigt die definierten Policies. Zeile 1 bereibt die Policies für eingehende Pakete. Diese werden mit ESP verschlüsselt und vom Verbindungspartner `213.150.1.160` versandt. Der innere IP Header enthält IP Pakete mit Sourceadressen aus dem Netz `192.168.1.0/24` und Destinationadressen aus dem Netz `192.168.2.0/24`.

Durch den schon oben erwähnten Parameter `active` in der Datei `/etc/ipsec.conf` werden die Security Associations sofort ausverhandelt. Diese können mit `ipsecctl -s sa` angezeigt werden:

Listing 7: SAD

```
1 esp tunnel from 213.150.1.161 to 213.150.1.160 spi 0x9488ba37 auth hmac-sha2-256 enc aes \
    authkey 0x56dc2d8271d91c51101b87cab09783012e80820f14d82040d904791c140554c5 \
3   enckey 0xe75e29171ce142cb9133bb6cff6c4ea0
  esp tunnel from 213.150.1.160 to 213.150.1.161 spi 0x0ceebee4 auth hmac-sha2-256 enc aes \
5   authkey 0x280203a6808011119801367 35fe553428c60dec100ad3152d11244a896422d72 \
    enckey 0xa887d148c59a85ba36c3b392d3bbfdfb
```

In Listing 7 sind zwei Security Associations dargestellt. Die Zeilen 1 bis 3 beschreiben eine SA, die einen IPSec Tunnel, der mit ESP gesichert wurde, definiert. Man erkennt den Authentifizierungsalgorithmus `hmac-sha2-256` und den Verschlüsselungsalgorithmus `Advanced Encryption Standard (AES)`. Zeile 2 enthält den Authentifizierungsschlüssel und in Zeile 3 ist der Verbindungsschlüssel abgebildet.

Zusätzlich muss noch in `/etc/sysctl.conf` das Weiterleiten von IP Paketen aktiviert werden.

Listing 8: Weiterleiten von IP Paketen via sysctl

```
net.ip.forwarding=1
```

Dies ist auch erforderlich, um den Clients im LAN1 und LAN2 eine Verbindung mit dem öffentlichen Netz zu ermöglichen. Es wurde auf beiden Servern Network Address Translation (NAT) [SE01] konfiguriert. OBSD1 verbindet LAN1 mit dem öffentlichen Netz, OBSD2 Clients im LAN2. Konfiguriert wird NAT unter OpenBSD durch folgende Einträge in `/etc/pf.conf`:

Listing 9: Firewallkonfiguration

```
1 ext_if=pcn0
  ipphone1=192.168.1.2
3 int_net=192.168.1.0/24

5 nat on $ext_if proto udp from $ipphone1 to any -> ($ext_if) static-port

7 pass out quick on $ext_if inet proto {tcp,udp} from $int_net to any keep state
```

Zeile 1 bewirkt, dass der Paketfilter in allen Paketen mit der Source IP-Adresse `192.168.1.2` eben diese Source IP Adresse auf die IP Adresse des externen Interfaces ändert. Auf dem Server OBSD1 wäre das zum Beispiel `213.150.1.161`. OpenBSD ändert normalerweise auch die Source Ports der UDP und TCP Pakete. Die IP Telefone müssen jedoch für eine funktionierende SIP Session in das öffentliche Netz ihre externe IP Adresse kennen. Dazu dient das Protokoll Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (STUN) [RWHM03]. Dieses Protokoll funktioniert allerdings nicht, wenn die Source Ports verändert werden. Der Eintrag `static-port` in Zeile 3 verhindert das Umschreiben der Ports.

Zeile 5 aktiviert eine sogenannte Stateful Inspection der TCP und UDP Pakete. Der Paketfilter merkt sich ausgehenden Verbindungen (Source IP, Source Port, Destination IP, Destination Port) und ermöglicht ein Antworten.

OpenBSD bietet auch ein Netzwerkinterface Namens `enc0`. Werden Pakete mit Hilfe des Tools `tcpdump(1)` auf diesem Netzwerkinterface aufgezeichnet, sind die verschlüsselten Daten im Klartext sichtbar. Dies ist möglich, da der OpenBSD Server als IPSec Gateway die privaten Schlüssel kennt.

4.1.2 OpenSER

OpenSER (`http://openser.org`, Stand 27. Februar 2007) ist ein frei verfügbarer SIP Proxy. Für die Teststellung wurde Version 1.1.0 verwendet. Unter OpenBSD musste die Version 1.1.0 aus dem CVS ausgecheckt werden, da die downloadbare tar.gz Datei einen Fehler enthält und sich nicht kompilieren ließ.

Als Datenbankbackend für OpenSER kam PostgreSQL (`http://postgresql.org`, Stand 27. Februar 2007) in der Version 8.1.5 zum Einsatz. Leider unterstützt OpenSER in der Version 1.1.0 PostgreSQL nicht unter OpenBSD. Der Patch im Anhang A ermöglicht das Kompilieren von OpenSER mit Postgres als Datenbankbackend unter OpenBSD. Damit sich der SIP Proxy ordnungsgemäß zur Datenbank verbinden kann, muss die maximale Anzahl der verfügbaren Semaphoren im Betriebssystem erhöht werden. Dies geschieht in der Datei `/etc/sysctl.conf`:

Listing 10: SEMMNS

```
1 kern.seminfo.semmns=120
```

Zusätzlich muss die Anzahl der möglichen Datenbankverbindungen erhöht werden. Dies geschieht unter dem User `_postgressql` in der Datei `data/postgres.conf`:

Listing 11: max_connections

```
1 max_connections = 80
```

Installiert wurde OpenSER auf dem Server OBSD1. Dadurch ist der SIP Proxy sowohl für die internen User Agents, als auch für Clients, die sich im öffentlichen Netz befinden, erreichbar.

Als Client wurde ein Windows Softphone verwendet (X-Lite Version 3.0 `http://www.xten.com`).

4.1.3 Netzwerkaufbau

In Abbildung 8 ist der Netzwerkaufbau der verwendeten Teststellung dargestellt.

Für die unter 4.2 beschriebenen Szenarien wurden SIP User Agents (Windows Softphone X-lite 3.0) in beiden privaten und im öffentlichen Netz aufgestellt. Die beiden privaten Netze wurden wie in 4.1.1.1 dargestellt, mit einem IPSec ESP Tunnel verbunden. Als SIP Proxy für die beiden UA in den internen Netzen dient OpenSER auf dem der Server OBSD1. Dieser Server verbindet auch die internen Clients im LAN1 mit dem öffentlichen Netzwerk. Dies geschieht durch Network Address Translation (NAT) (siehe 4.1.1.1). Der Server OBSD2 verbindet durch NAT die Clients im LAN2 mit dem öffentlichen Netz.

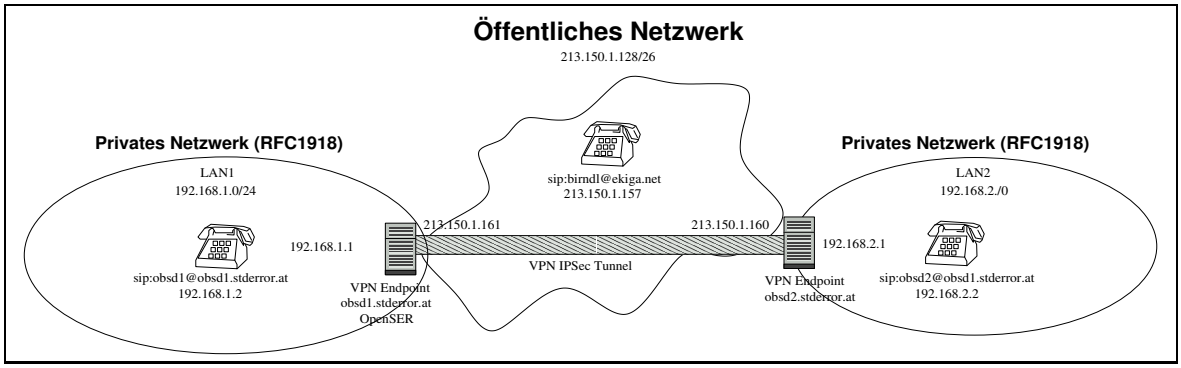

Abb. 8: Netzwerkaufbau

Der UA im öffentlichen Netz bedient sich eines frei verfügbaren SIP Proxies (`http://ekiga.org`).

4.2 Durchgeführte Tests und deren Ergebnisse

In den folgenden Kapiteln werden die getesteten Szenarien und deren Ergebnisse beschrieben. Nach einer Erklärung der durchgeführten Tests im Abschnitt `Aufbau`, folgen die Ergebnisse im Abschnitt `Ergebnisse`. Zuerst wurde eine LAN zu LAN Verbindung getestet. Danach folgte ein Verbindungstest des Clients im LAN `192.168.2.0/24` zu einem Client, der sich im öffentlichen Netzwerk befand. Abschließend wurde ein Anruf des öffentlichen Clients zum Client im LAN `192.168.2.0/24` untersucht.

4.2.1 SIP Session LAN1 zu LAN2

Hier wurde eine SIP-Session zwischen LAN1 und LAN2 analysiert. Es sollte untersucht werden, ob die Signalisierung und der Mediastream unterschiedliche Wege im Netzwerk nehmen. Der User Agent mit der Adresse `192.168.1.2` baut dazu eine SIP Session zu dem User Agent mit der Adresse `192.168.2.2` auf.

4.2.1.1 Aufbau

Zuerst wurde ein Verbindungsaufbau von UA1 zu UA2 simuliert. Auf einen Test in Gegenrichtung wurde verzichtet, da hier keine Unterschiede beim Verlauf der Signalisierung oder des Mediastreams im Netzwerk zu erwarten sind. Zuerst registieren sich beide User Agents am SIP Proxy, danach folgt der Aufbau der Session von UA1 zu UA2.

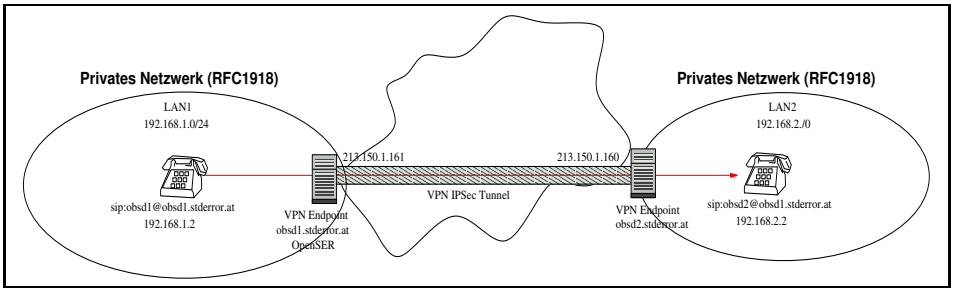

Abb. 9: Kommunikation LAN1 - LAN2

4.2.1.2 Ergebnisse

Die Registrierung des UA1 am SIP-Proxy erfolgt im lokalen LAN1, dies hat keine Auswirkungen auf den bestehenden IPSec Tunnel.

Listing 12: Register Request UA1 obsd1@192.168.1.2

```
1  Internet Protocol, Src: 192.168.1.2 (192.168.1.2), Dst: 192.168.1.1 (192.168.1.1)
   User Datagram Protocol, Src Port: 23416 (23416), Dst Port: 5060 (5060)
3  Session Initiation Protocol
       Request-Line: REGISTER sip:obsd1.stderror.at SIP/2.0
5      Message Header
           Via: SIP/2.0/UDP 192.168.1.2:23416;branch=z9hG4bK-d87543-0d5b2a6fc544322c-1--d87543-;rport
7          Max-Forwards: 70
           Contact: <sip:obsd1@192.168.1.2:23416;rinstance=9b04cb80691af9f2>
9          To: "obsd1"<sip:obsd1@obsd1.stderror.at>
           From: "obsd1"<sip:obsd1@obsd1.stderror.at>;tag=4c2e1e50
11         Call-ID: NDAzZDFiY2EwZmNhMjEyYmZjZWVjZmRjZjY2ZjZhMDM.
           CSeq: 2 REGISTER
13         Expires: 3600
           Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE, SUBSCRIBE, INFO
15         User-Agent: X-Lite release 1006e stamp 34025
           Content-Length: 0
```

Listing 12 zeigt, dass der Register Request direkt vom User Agent 1 mit der Adresse `192.168.1.2` an den SIP Proxy mit der Adresse `192.168.1.1` versandt wird (Zeile1). Es erfolgt daher keine Kommunikation über die mit IPSec gesicherte Verbindung. Als Contact Adresse verwendet das Softphone die konfigurierte Adresse im LAN1 (Zeile 8).

Listing 13: Register Request UA2 obsd2@192.168.2.2

```
   Enc IPv4, SPI 0xa43e4f41, authentic, confidential
2  Internet Protocol, Src: 213.150.1.160 (213.150.1.160), Dst: 213.150.1.161 (213.150.1.161)
   Internet Protocol, Src: 192.168.2.2 (192.168.2.2), Dst: 192.168.1.1 (192.168.1.1)
4  User Datagram Protocol, Src Port: 37342 (37342), Dst Port: 5060 (5060)
   Session Initiation Protocol
6      Request-Line: REGISTER sip:obsd1.stderror.at SIP/2.0
       Message Header
8          Via: SIP/2.0/UDP 192.168.2.2:37342;branch=z9hG4bK-d87543-c13b4e4b990b7718-1--d87543-;rport
           Max-Forwards: 70
10         Contact: <sip:obsd2@192.168.2.2:37342;rinstance=04748e01b6c0771d>
           To: "obsd2"<sip:obsd2@obsd1.stderror.at>
12         From: "obsd2"<sip:obsd2@obsd1.stderror.at>;tag=e65e9f53
           Call-ID: M2FjYzAwZTg3MzE1YzYwYWRmNWY4N2IxY2UxZDcxMTk.
14         CSeq: 2 REGISTER
           Expires: 3600
16         Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE, SUBSCRIBE, INFO
           User-Agent: X-Lite release 1006e stamp 34025
18         Content-Length: 0
```

In Listing 13 ist der Register Request des User Agents 2 an den SIP Proxy angeführt. Zeile 1 lässt erkennen, dass es sich um ein verschlüsseltes Paket (`Enc`) mit dem Security Parameter Index (SPI, vergleiche 3.2.4) `0xa43e4f41` handelt. Der äußere IP Header des ESP Paketes enthält die öffentlichen Absenderadresse des Servers OBSD1 und als Empfänger die öffentliche Adresse des Server OBSD2. Der innere IP Header enthält die ursprünglichen

Adressen des User Agents UA2 und des SIP Proxies. Als Contact Adresse dient die Adresse des User Agent im LAN2 `192.168.2.2`.

Ruft nun der User Agent 1 den User Agent 2 an, wird von UA1 folgender `INVITE` Request generiert:

Listing 14: Invite Request UA1 -> SIP-Proxy

```
Internet Protocol, Src: 192.168.1.2 (192.168.1.2), Dst: 192.168.1.1 (192.168.1.1)
User Datagram Protocol, Src Port: 23416 (23416), Dst Port: 5060 (5060)
Session Initiation Protocol
    Request-Line: INVITE sip:obsd2@obsd1.stderror.at SIP/2.0
    Message Header
        Via: SIP/2.0/UDP 192.168.1.2:23416;branch=z9hG4bK-d87543-5f0c1627b5382207-1--d87543-;rport
        Max-Forwards: 70
        Contact: <sip:obsd1@192.168.1.2:23416>
        To: "obsd2@obsd1.stderror.at"<sip:obsd2@obsd1.stderror.at>
        From: "obsd1"<sip:obsd1@obsd1.stderror.at>;tag=7938f312
        Call-ID: MzMyNGEyMWQ1M2ZjMWVkMTFjYzQwNDQyNGNiMjExNWE.
        CSeq: 1 INVITE
        Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE, SUBSCRIBE, INFO
        Content-Type: application/sdp
        User-Agent: X-Lite release 1006e stamp 34025
        Content-Length: 273
    Message body
        Session Description Protocol
            Session Description Protocol Version (v): 0
            Owner/Creator, Session Id (o): - 0 2 IN IP4 192.168.1.2
            Session Name (s): CounterPath X-Lite 3.0
            Connection Information (c): IN IP4 192.168.1.2
            Time Description, active time (t): 0 0
            Media Description, name and address (m): audio 24014 RTP/AVP 107 119 0 98 8 3 101
            Media Attribute (a): fmtp:101 0-15
            Media Attribute (a): rtpmap:107 BV32/16000
            Media Attribute (a): rtpmap:119 BV32-FEC/16000
            Media Attribute (a): rtpmap:98 iLBC/8000
            Media Attribute (a): rtpmap:101 telephone-event/8000
            Media Attribute (a): sendrecv
```

Listing 14 zeigt einen Invite Request. Absender ist der UA1 mit der internen Adresse `192.168.1.2`, Empfänger ist der SIP Proxy mit der internen Adresse `192.168.1.1` Es wird der Address of Record (AoR) `obsd2@osbd1.stderror.at` (Zeile 8) verwendet. Als Contact Adresse dient `obsd1@192.168.1.2` (Zeile 10). Im Session Description Protokoll (SDP) [HJP06] ist die interne Adresse des User Agents angegeben `192.168.1.2` (Zeile 22).

Das Paket wird vom SIP Proxy an den UA2 weitergereicht:

Listing 15: Invite Request Proxy -> UA2

```
Enc IPv4, SPI 0xa7013c07, authentic, confidential
Internet Protocol, Src: 213.150.1.161 (213.150.1.161), Dst: 213.150.1.160 (213.150.1.160)
Internet Protocol, Src: 192.168.1.1 (192.168.1.1), Dst: 192.168.2.2 (192.168.2.2)
User Datagram Protocol, Src Port: 5060 (5060), Dst Port: 37342 (37342)
Session Initiation Protocol
    Request-Line: INVITE sip:obsd2@192.168.2.2:37342;rinstance=04748e01b6c0771d SIP/2.0
```

Das Paket wird durch den ESP Tunnel an den User Agent 2 weitergereicht (Zeile 1, 2 und 3). Der Proxy hat den `INVITE` Request modifiziert. In Zeile 6 wird als SIP URL `sip:obsd@192.168.2.2` angegeben. Beim vorhergehenden Paket von UA1 zum Proxy wurde noch `sip:obsd@obsd1.stderror.at` als URL benutzt.

Die Antwort `200 OK` des UA2 erfolgt wieder über die gesicherte Verbindung:

Listing 16: OK Request UA2 -> Proxy

```
Enc IPv4, SPI 0xa43e4f41, authentic, confidential
Internet Protocol, Src: 213.150.1.160 (213.150.1.160), Dst: 213.150.1.161 (213.150.1.161)
Internet Protocol, Src: 192.168.2.2 (192.168.2.2), Dst: 192.168.1.1 (192.168.1.1)
User Datagram Protocol, Src Port: 37342 (37342), Dst Port: 5060 (5060)
Session Initiation Protocol
    Status-Line: SIP/2.0 200 OK
        Status-Code: 200
        [Resent Packet: False]
    Message Header
        Via: SIP/2.0/UDP 192.168.1.1;branch=z9hG4bK5bb8.8e6dcb97.0
        Via: SIP/2.0/UDP 192.168.1.2:23416;branch=z9hG4bK-d87543-5f0c1627b5382207-1--d87543-;rport=23416
```

```
12          Record-Route: <sip:192.168.1.1;lr;ftag=7938f312>
            Contact: <sip:obsd2@192.168.2.2:37342;rinstance=04748e01b6c0771d>
14          To: "obsd2@obsd1.stderror.at"<sip:obsd2@obsd1.stderror.at>;tag=fc664f25
            From: "obsd1"<sip:obsd1@obsd1.stderror.at>;tag=7938f312
16          Call-ID: MzMyNGEyMWQ1M2ZjMWVkMTFjYzQwNDQyNGNiMjExNWE.
            CSeq: 1 INVITE
18          Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE, SUBSCRIBE, INFO
            Content-Type: application/sdp
20          User-Agent: X-Lite release 1006e stamp 34025
            Content-Length: 273
22      Message body
            Session Description Protocol
24              Session Description Protocol Version (v): 0
                Owner/Creator, Session Id (o): - 8 2 IN IP4 192.168.2.2
26              Session Name (s): CounterPath X-Lite 3.0
                Connection Information (c): IN IP4 192.168.2.2
28              Time Description, active time (t): 0 0
                Media Description, name and address (m): audio 22546 RTP/AVP 107 119 0 98 8 3 101
30              Media Attribute (a): fmtp:101 0-15
                Media Attribute (a): rtpmap:107 BV32/16000
32              Media Attribute (a): rtpmap:119 BV32-FEC/16000
                Media Attribute (a): rtpmap:98 iLBC/8000
34              Media Attribute (a): rtpmap:101 telephone-event/8000
                Media Attribute (a): sendrecv
```

Es wurde wieder der Tunnel zwischen den beiden Netzen zur Übertragung genutzt (Zeilen 1-3). Der User Agent antwortet mit einem `200 OK` auf den vorgehenden `INVITE` Request. Im SDP Teil findet sich die Adressen des User Agents 2 wieder (`192.168.2.2`). Es sind jetzt beiden Endgeräten die Kontaktdaten (IP Adresse und UDP Port) für den Mediastream bekannt.

Listing 17: Media Stream UA1 <-> UA2

```
1   Enc IPv4, SPI 0xa7013c07, authentic, confidential
    Internet Protocol, Src: 213.150.1.161 (213.150.1.161), Dst: 213.150.1.160 (213.150.1.160)
3   Internet Protocol, Src: 192.168.1.2 (192.168.1.2), Dst: 192.168.2.2 (192.168.2.2)
    User Datagram Protocol, Src Port: 24014 (24014), Dst Port: 22546 (22546)
5   Real-Time Transport Protocol

7   Enc IPv4, SPI 0xa43e4f41, authentic, confidential
    Internet Protocol, Src: 213.150.1.160 (213.150.1.160), Dst: 213.150.1.161 (213.150.1.161)
9   Internet Protocol, Src: 192.168.2.2 (192.168.2.2), Dst: 192.168.1.2 (192.168.1.2)
    User Datagram Protocol, Src Port: 22546 (22546), Dst Port: 24014 (24014)
11  Real-Time Transport Protocol
```

Listing 17 stellt zwei Pakete aus dem Media Steam Beispiel dar. Beide Pakete werden über den geschützten Kanal übertragen (Zeilen 1-3 und 7-9).

Eine eventuell zu erwartende Verzögerung der Registrierung des UA2 tritt nicht auf, da der OpenBSD ISAKMP Daemon sofort nach Aufruf des Befehls `ipsectl -f /etc/ipsec.conf` mit dem Ausverhandeln der Security Associations beginnt. Dies ist insofern bemerkenswert, da der Autor bereits Erfahrungen mit IPSec Gateway's unter FreeBSD (`http://www.freebsd.org`, Stand 27. Februar 2007) hat. Dort werden die SA's erst dann ausverhandelt, wenn das erste Paket über den konfigurierten Tunnel versandt wird. Dies führt zu einer Verzögerung von einigen Sekunden.

4.2.2 SIP Session LAN1 zu LAN2 mit STUN

Im vorhergehenden Test 4.2.1 ist es den beiden User Agents nicht möglich, einen Verbindungsaufbau zu einem User Agent im öffentlichen Netz aufzubauen. Ein Sessionaufbau ist nur zwischen LAN1 und LAN2 möglich. Wollen User Agent 1 oder User Agent 2 aber auch Teilnehmer im öffentlichen Netz erreichen, müssen sie ihre eigene externe IP-Adresse kennen, da sich im SDP Teil die Kontaktdaten (IP Adresse und TCP Port) für den Remote User Agent befinden. Würden die internen Adressen verwendet werden, die im öffentlich Netz

nicht erreichbar sind, kann ein SDP Verbindungsaufbau nicht erfolgen. Dazu muss der User Agent im internen LAN seine externe IP Adresse im SDP Teil des SIP Paketes eintragen. Zum Erkennen der externen Adresse muss STUN [RWHM03] am Client aktiviert werden. Da sich dies aber auch auf den Verlauf der Verbindung bei einer Session zwischen LAN1 und LAN2 auswirken kann, wurde der Test aus Kapitel 4.2.1 mit aktiviertem STUN wiederholt. Bei beiden SIP Telefonen wurde der öffentliche STUN Server `stun.ekiga.org` eingetragen.

4.2.2.1 Aufbau

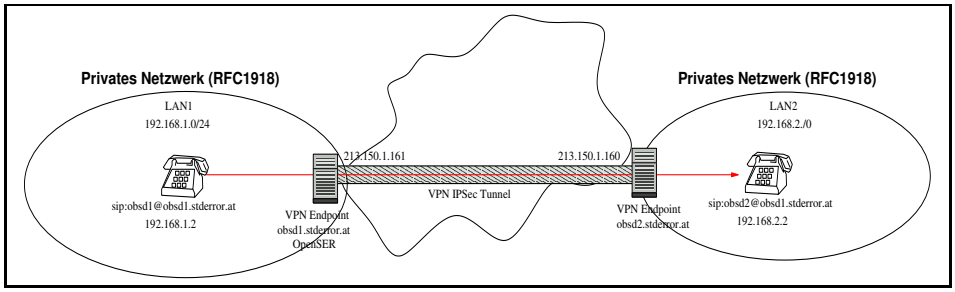

Abb. 10: Kommunikation LAN1 - LAN2

Wieder ruft der User Agent in LAN1 den User Agent in LAN2 an. Es wurde untersucht, welche Wege der Sessionaufbau und der Mediastream im Netzwerk nehmen. Beide User Agents registrieren sich am SIP Proxy, danach erfolgt der Verbindungsaufbau von UA1 zu UA2. Auf einen Test in Gegenrichtung wurde wie in 4.2.1 verzichtet, da keine Unterschiede zu erwarten sind.

4.2.2.2 Ergebnisse

Nach dem Starten des Softphones versucht dieses sofort, via STUN seine externe IP Adresse herauszufinden:

Listing 18: STUN Request obsd1@obsd1.stderror.at

```
1  Internet Protocol, Src: 192.168.1.2 (192.168.1.2), Dst: 194.39.182.241 (194.39.182.241)
   User Datagram Protocol, Src Port: 12186 (12186), Dst Port: 3478 (3478)
3  Simple Traversal of UDP Through NAT
       Message Type: Binding Request (0x0001)
5      Message Length: 0x0000
       Message Transaction ID: EC451ADB3120E748804BA3E21C0257B7
7
   Internet Protocol, Src: 194.39.182.241 (194.39.182.241), Dst: 192.168.1.2 (192.168.1.2)
9  User Datagram Protocol, Src Port: 3478 (3478), Dst Port: 12186 (12186)
   Simple Traversal of UDP Through NAT
11     Message Type: Binding Response (0x0101)
       Message Length: 0x0044
13     Message Transaction ID: EC451ADB3120E748804BA3E21C0257B7
       Attributes
```

Listing 18 zeigt einen Auszug der STUN Requests. Eine genau Erklärung des Protokolls findet sich in RFC3489 [RWHM03]. Dem Softphone ist nun die externe IP Adresse `213.150.1.161` bekannt. Es folgt die Registrierung des UA1 am SIP-Proxy. Dies erfolgt im lokalen LAN1 und hat keine Auswirkungen auf den bestehenden IPSec Tunnel.

Listing 19: Register Request UA1 obsd1@213.150.1.161

```
Internet Protocol, Src: 192.168.1.2 (192.168.1.2), Dst: 192.168.1.1 (192.168.1.1)
User Datagram Protocol, Src Port: 53352 (53352), Dst Port: 5060 (5060)
Session Initiation Protocol
    Request-Line: REGISTER sip:obsd1.stderror.at SIP/2.0
    Message Header
        Via: SIP/2.0/UDP 192.168.1.2:53352;branch=z9hG4bK-d87543-8b16d82be022002b-1--d87543-;rport
        Max-Forwards: 70
        Contact: <sip:obsd1@213.150.1.161:53352;rinstance=781884bd98dadbbb>;expires=0
        To: "obsd1"<sip:obsd1@obsd1.stderror.at>
        From: "obsd1"<sip:obsd1@obsd1.stderror.at>;tag=7008b71a
        Call-ID: NzhjM2RhNjFiYzNjYzI5YWM2YmJhZjdjMjQ1OTk5MTc.
        CSeq: 3 REGISTER
```

Es fällt auf, dass als Contact Adresse (Zeile 8, Listing 19) `obsd@213.150.1.161` angegeben wird. Obwohl beim Softphone UA1 die interne Adresse `192.168.1.1` als Outbound Proxy angegeben wurde, ermittelt das Telefon via STUN seine externe IP-Adresse. Es erfolgt ein zweiter Register Request, diesmal mit `obsd1@192.168.1.2` als Contact Adresse:

Listing 20: Register Request UA1 obsd1@192.168.1.2

```
Internet Protocol, Src: 192.168.1.2 (192.168.1.2), Dst: 192.168.1.1 (192.168.1.1)
User Datagram Protocol, Src Port: 53352 (53352), Dst Port: 5060 (5060)
Session Initiation Protocol
    Request-Line: REGISTER sip:obsd1.stderror.at SIP/2.0
    Message Header
        Via: SIP/2.0/UDP 192.168.1.2:53352;branch=z9hG4bK-d87543-851f22547b720439-1--d87543-;rport
        Max-Forwards: 70
        Contact: <sip:obsd1@192.168.1.2:53352;rinstance=48f5a241c46ebb37>
        To: "obsd1"<sip:obsd1@obsd1.stderror.at>
        From: "obsd1"<sip:obsd1@obsd1.stderror.at>;tag=7008b71a
        Call-ID: NzhjM2RhNjFiYzNjYzI5YWM2YmJhZjdjMjQ1OTk5MTc.
        CSeq: 4 REGISTER
```

In Zeile 8 findet sich die Contact Adresse.

UA2 stellt ebenfalls zuerst via STUN seine externe IP Adresse fest (vergleiche Listing 18). Danach folgt die Registrierung am SIP-Proxy. Es laufen alle Signalisierungsdaten wie erwartet über die abgesicherte IPSec Verbindung.

Listing 21: Register Request UA2 obsd2@213.150.1.160

```
Enc IPv4, SPI 0xcf528764, authentic, confidential
Internet Protocol, Src: 213.150.1.160 (213.150.1.160), Dst: 213.150.1.161 (213.150.1.161)
Internet Protocol, Src: 192.168.2.2 (192.168.2.2), Dst: 192.168.1.1 (192.168.1.1)
User Datagram Protocol, Src Port: 17494 (17494), Dst Port: 5060 (5060)
Session Initiation Protocol
    Request-Line: REGISTER sip:obsd1.stderror.at SIP/2.0
    Message Header
        Via: SIP/2.0/UDP 192.168.2.2:17494;branch=z9hG4bK-d87543-0a678d605442ca50-1--d87543-;rport
        Max-Forwards: 70
        Contact: <sip:obsd2@213.150.1.160:17494;rinstance=93f7efb8b4386323>
        To: "obsd2"<sip:obsd2@obsd1.stderror.at>
        From: "obsd2"<sip:obsd2@obsd1.stderror.at>;tag=eb715256
        Call-ID: YzM2OGE0N2RmYzRmYTg1YzQyMjkyYTY0N2NiODQzYmM.
        CSeq: 1 REGISTER
```

Zeile 1 in Listing 21 lässt erkennen, dass es sich um ein verschlüsseltes Paket (Enc) mit dem Security Parameter Index (SPI, vergleiche 3.2.4) `0xcf528764` handelt. Der äußere IP Header des ESP Paketes enthält die öffentliche Absender Adresse des Server OBSD1 und als Empfänger die öffentliche Empfangsadresse des Servers OBSD2. Der innere IP Header enthält die ursprünglichen Adressen des User Agents UA2 und des SIP Proxies. Auch hier wurde vom Softphone durch STUN die externe IP-Adresse `213.150.1.160` ermittelt. Diese wird dann auch als Contact Adresse (Zeile 10) `obsd@213.150.1.160` beim Register Request angegeben. Es erfolgt ein zweiter Register Request mit `obsd2@192.168.2.2` als Contact Adresse:

Listing 22: Register Request UA2 obsd2@192.168.2.2

```
Enc IPv4, SPI 0xcf528764, authentic, confidential
Internet Protocol, Src: 213.150.1.160 (213.150.1.160), Dst: 213.150.1.161 (213.150.1.161)
```

```
Internet Protocol, Src: 192.168.2.2 (192.168.2.2), Dst: 192.168.1.1 (192.168.1.1)
User Datagram Protocol, Src Port: 17494 (17494), Dst Port: 5060 (5060)
Session Initiation Protocol
    Request-Line: REGISTER sip:obsd1.stderror.at SIP/2.0
    Message Header
        Via: SIP/2.0/UDP 192.168.2.2:17494;branch=z9hG4bK-d87543-ad686a7cec63fe50-1--d87543-;rport
        Max-Forwards: 70
        Contact: <sip:obsd2@192.168.2.2:17494;rinstance=137e823211727c17>
        To: "obsd2"<sip:obsd2@obsd1.stderror.at>
        From: "obsd2"<sip:obsd2@obsd1.stderror.at>;tag=eb715256
        Call-ID: YzM2OGE0N2RmYzRmYTg1YzQyMjkyYTY0N2NiODQzYmM.
        CSeq: 4 REGISTER
```

Es handelt sich wieder um ein verschlüsseltes IP Paket (Enc, Zeile 1). Der äußere IP Header enthält die öffentliche IP Adresse des Servers OBSD2 als Absender und die öffentliche IP Adresse des Servers OBSD1 als Empfänger. Der innere IP Header besteht aus der internen Adresse des UA2 als Absender und der internen Adresse des SIP Proxies als Empfänger. In Zeile 10 ist wieder die geänderte Contact Adresse ersichtlich.

Ruft nun UA1 UA2 an, wird von UA1 folgender `INVITE` Request generiert:

Listing 23: Invite Request UA1 -> Proxy

```
Internet Protocol, Src: 192.168.1.2 (192.168.1.2), Dst: 192.168.1.1 (192.168.1.1)
User Datagram Protocol, Src Port: 53352 (53352), Dst Port: 5060 (5060)
Session Initiation Protocol
    Request-Line: INVITE sip:obsd2@obsd1.stderror.at SIP/2.0
    Message Header
        Via: SIP/2.0/UDP 192.168.1.2:53352;branch=z9hG4bK-d87543-42329a31c47e4375-1--d87543-;rport
        Max-Forwards: 70
        Contact: <sip:obsd1@192.168.1.2:53352>
        To: "obsd2@obsd1.stderror.at"<sip:obsd2@obsd1.stderror.at>
        From: "obsd1"<sip:obsd1@obsd1.stderror.at>;tag=c57d365dn
        Call-ID: NTIwNDQ0OTVlOTVjYzI0NGFlNGVmYTQwNjIwOTkwOWE.
        CSeq: 1 INVITE
        Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE, SUBSCRIBE, INFO
        Content-Type: application/sdp
        User-Agent: X-Lite release 1006e stamp 34025
        Content-Length: 479
    Message body
        Session Description Protocol
            Session Description Protocol Version (v): 0
            Owner/Creator, Session Id (o): - 8 2 IN IP4 213.150.1.161
            Session Name (s): CounterPath X-Lite 3.0
            Connection Information (c): IN IP4 213.150.1.161
            Time Description, active time (t): 0 0
            Media Description, name and address (m): audio 38686 RTP/AVP 107 119 0 98 8 3 101
            Media Attribute (a): alt:1 4 : 6GuWKy8b rEoWTR6H 192.168.1.2 38686
            Media Attribute (a): alt:2 3 : H0qnXKZ4 ywZ2rnkb 192.168.181.1 38686
            Media Attribute (a): alt:3 2 : bcITwweS dsxAKlGu 192.168.154.1 38686
            Media Attribute (a): alt:4 1 : CtvgmZt3 AHh4RNJ4 213.150.1.161 38686
```

Absender ist der UA1 mit der internen Adresse `192.168.1.2`, Empfänger ist der SIP Proxy mit der internen Adresse `192.168.1.1` Es wird der Address of Record (AoR) `obsd2@osbd1.stderror.at` (Zeile 8) verwendet. Als Contact Adresse dient `obsd1@192.168.1.2` (Zeile 10). Interessant ist hier, dass im SDP Teil [HJP06] die öffentliche Adresse angegeben wird `213.150.1.161` (Zeile 22). Diese wurde durch STUN vom Client vor dem Verbindungsaufbau erkannt.

Das Paket wird vom SIP Proxy an den UA2 weitergereicht:

Listing 24: Invite Request Proxy -> UA2

```
Enc IPv4, SPI 0xd7d5bb2c, authentic, confidential
Internet Protocol, Src: 213.150.1.161 (213.150.1.161), Dst: 213.150.1.160 (213.150.1.160)
Internet Protocol, Src: 192.168.1.1 (192.168.1.1), Dst: 192.168.2.2 (192.168.2.2)
User Datagram Protocol, Src Port: 5060 (5060), Dst Port: 17494 (17494)
Session Initiation Protocol
    Request-Line: INVITE sip:obsd2@192.168.2.2:17494;rinstance=137e823211727c17 SIP/2.0
```

Das Paket wird durch den ESP Tunnel an den User Agent 2 weitergereicht (Zeile 1, 2 und 3). Der Proxy hat den `INVITE` Request modifiziert. In Zeile 6 wird als SIP URL `sip:obsd@192.168.2.2` angegeben. Beim vorhergehenden Paket von UA1 zum Proxy wurde noch `sip:obsd@obsd1.stderror.at` als URL benutzt.

UA2 verschickt als nächstes einen STUN Binding Request. Daran erkennt man, dass gewisse User Agent autonom agieren.

Listing 25: STUN Binding Request UA2 -> UA1

```
Enc IPv4, SPI 0xcf528764, authentic, confidential
Internet Protocol, Src: 213.150.1.160 (213.150.1.160), Dst: 213.150.1.161 (213.150.1.161)
Internet Protocol, Src: 192.168.2.2 (192.168.2.2), Dst: 192.168.1.2 (192.168.1.2)
User Datagram Protocol, Src Port: 13780 (13780), Dst Port: 38686 (38686)
Simple Traversal of UDP Through NAT
    Message Type: Binding Request (0x0001)
    Message Length: 0x0024
    Message Transaction ID: 7A939872A01E7E46BC54DC82F5F88716
    Attributes
        Attribute: USERNAME
        Attribute: MESSAGE-INTEGRITY
```

UA1 antwortet mit einem STUN Binding Response:

Listing 26: STUN Response UA1 -> UA2

```
Enc IPv4, SPI 0xd7d5bb2c, authentic, confidential
Internet Protocol, Src: 213.150.1.161 (213.150.1.161), Dst: 213.150.1.160 (213.150.1.160)
Internet Protocol, Src: 192.168.1.2 (192.168.1.2), Dst: 192.168.2.2 (192.168.2.2)
User Datagram Protocol, Src Port: 38686 (38686), Dst Port: 13780 (13780)
Simple Traversal of UDP Through NAT
    Message Type: Binding Response (0x0101)
    Message Length: 0x0048
    Message Transaction ID: 7A939872A01E7E46BC54DC82F5F88716
    Attributes
        Attribute: MAPPED-ADDRESS
            Attribute Type: MAPPED-ADDRESS (0x0001)
            Attribute Length: 8
            Protocol Family: IPv4 (0x0001)
            Port: 13780
            IP: 192.168.2.2 (192.168.2.2)
        Attribute: CHANGED-ADDRESS
            Attribute Type: CHANGED-ADDRESS (0x0005)
            Attribute Length: 8
            Protocol Family: IPv4 (0x0001)
            Port: 38686
            IP: 192.168.1.2 (192.168.1.2)
        Attribute: SOURCE-ADDRESS
            Attribute Type: SOURCE-ADDRESS (0x0004)
            Attribute Length: 8
            Protocol Family: IPv4 (0x0001)
            Port: 38686
            IP: 192.168.1.2 (192.168.1.2)
```

Die beiden Softphones kennen durch dieses Verhalten nun auch ihre internen Adressen und wissen, dass eine Kommunikation auch direkt möglich ist.

Die Antwort `200 OK` des UA2 erfolgt wieder über die gesicherte Verbindung:

Listing 27: OK Request UA2 -> Proxy

```
Enc IPv4, SPI 0xcf528764, authentic, confidential
Internet Protocol, Src: 213.150.1.160 (213.150.1.160), Dst: 213.150.1.161 (213.150.1.161)
Internet Protocol, Src: 192.168.2.2 (192.168.2.2), Dst: 192.168.1.1 (192.168.1.1)
User Datagram Protocol, Src Port: 17494 (17494), Dst Port: 5060 (5060)
Session Initiation Protocol
    Status-Line: SIP/2.0 200 OK
    Message Header
        Via: SIP/2.0/UDP 192.168.1.1;branch=z9hG4bKce51.60c7fdd1.0
        Via: SIP/2.0/UDP 192.168.1.2:53352;branch=z9hG4bK-d87543-42329a31c47e4375-1--d87543-;rport=53352
        Record-Route: <sip:192.168.1.1;lr;ftag=c57d365d>
        Contact: <sip:obsd2@192.168.2.2:17494;rinstance=137e823211727c17>
        To: "obsd2@obsd1.stderror.at"<sip:obsd2@obsd1.stderror.at>;tag=43369163
        From: "obsd1"<sip:obsd1@obsd1.stderror.at>;tag=c57d365d
        Call-ID: NTIwNDQ0OTVlOTVjYzI0NGFlNGVmYTQwNjIwOTkwOWE.
        CSeq: 1 INVITE
        Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE, SUBSCRIBE, INFO
        Content-Type: application/sdp
        User-Agent: X-Lite release 1006e stamp 34025
        Content-Length: 377
    Message body
        Session Description Protocol
            Session Description Protocol Version (v): 0
            Owner/Creator, Session Id (o): - 1 2 IN IP4 213.150.1.160
            Session Name (s): CounterPath X-Lite 3.0
            Connection Information (c): IN IP4 213.150.1.160
            Time Description, active time (t): 0 0
            Media Description, name and address (m): audio 13780 RTP/AVP 107 119 0 98 8 3 101
            Media Attribute (a): alt:1 2 : ldGJzJ90 jRHc0V8n 192.168.2.2 13780
            Media Attribute (a): alt:2 1 : 1QXzOkAN S/UQwibc 213.150.1.160 13780
```

Auch hier wird im SDP Teil die externe IP-Adresse des UA2 angegeben (Zeile 25).

Nun folgt der Mediastream zwischen den beiden User Agents. Obwohl beim initialen `INVITE` als auch beim `200 OK` im SDP Teil die externen IP Adressen angegeben werden, verläuft die Media Verbindung **nicht** über das öffentliche Netz, sondern über die IPSec Verbindung.

Listing 28: Media Stream UA1 <-> UA2

```
1  Enc IPv4, SPI 0xd7d5bb2c, authentic, confidential
   Internet Protocol, Src: 213.150.1.161 (213.150.1.161), Dst: 213.150.1.160 (213.150.1.160)
3  Internet Protocol, Src: 192.168.1.2 (192.168.1.2), Dst: 192.168.2.2 (192.168.2.2)
   User Datagram Protocol, Src Port: 38686 (38686), Dst Port: 13780 (13780)
5
   Enc IPv4, SPI 0xcf528764, authentic, confidential
7  Internet Protocol, Src: 213.150.1.160 (213.150.1.160), Dst: 213.150.1.161 (213.150.1.161)
   Internet Protocol, Src: 192.168.2.2 (192.168.2.2), Dst: 192.168.1.2 (192.168.1.2)
9  User Datagram Protocol, Src Port: 13780 (13780), Dst Port: 38686 (38686)
```

Listing 28 stellt zwei Pakete aus dem Media Steam Beispiel dar. Beide Pakete werden über den geschützten Kanal übertragen.

4.2.3 Aufbau einer SIP Session LAN2 zu Internet

Der UA im LAN2 ruft einen Teilnehmer im öffentlichen Netz an. Abschnitt 4.2.3.1 `Aufbau` beschreibt den prinzipiellen Ablauf. In 4.2.3.2 werden die Ergebnisse dargestellt.

4.2.3.1 Aufbau

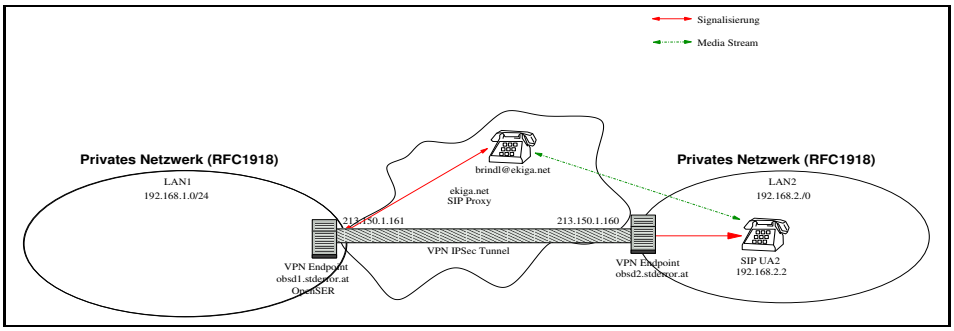

Abb. 11: Kommunikation LAN2 - öffentliches Netz

In Abbildung 11 ist der Session Aufbau eines SIP Calls von UA2, der sich im internen LAN2 befindet, und einem User Agent im öffentlichen Netz dargestellt. Es wird vorerst vermutet, dass der Signalisierungsverkehr der Session und der Media Streams unterschiedliche Wege im Netzwerk nehmen.

4.2.3.2 Ergebnisse

UA2 registriert sich wie in Abschnitt 4.2.1.2 beschrieben am SIP Proxy `obsd1.stderror.at` über die VPN Verbindung. Danach folgt der Invite Request:

Listing 29: INVITE Request UA2 -> Outbound Proxy

```
1  Enc IPv4, SPI 0xcf528764, authentic, confidential
   Internet Protocol, Src: 213.150.1.160 (213.150.1.160), Dst: 213.150.1.161 (213.150.1.161)
3  Internet Protocol, Src: 192.168.2.2 (192.168.2.2), Dst: 192.168.1.1 (192.168.1.1)
   User Datagram Protocol, Src Port: 13694 (13694), Dst Port: 5060 (5060)
5  Session Initiation Protocol
       Request-Line: INVITE sip:birndl@ekiga.net SIP/2.0
7      Message Header
           Via: SIP/2.0/UDP 192.168.2.2:13694;branch=z9hG4bK-d87543-275a9a5f121a365b-1--d87543-;rport
9          Max-Forwards: 70
           Contact: <sip:obsd2@192.168.2.2:13694>
11         To: "birndl@ekiga.net"<sip:birndl@ekiga.net>
           From: "obsd2"<sip:obsd2@obsd1.stderror.at>;tag=0a320a7d
13         CSeq: 1 INVITE
       Message body
15         Session Description Protocol
               Session Description Protocol Version (v): 0
17             Owner/Creator, Session Id (o): - 7 2 IN IP4 213.150.1.160
               Session Name (s): CounterPath X-Lite 3.0
19             Connection Information (c): IN IP4 213.150.1.160
               Time Description, active time (t): 0 0
21             Media Description, name and address (m): audio 18270 RTP/AVP 107 119 0 98 8 3 101
               Media Attribute (a): alt:1 2 : 6NTaFgsE qg21J1hm 192.168.2.2 18270
23             Media Attribute (a): alt:2 1 : e0fXDTlM taGSP03W 213.150.1.160 18270
               Media Attribute (a): fmtp:101 0-15
25             Media Attribute (a): rtpmap:107 BV32/16000
               Media Attribute (a): rtpmap:119 BV32-FEC/16000
27             Media Attribute (a): rtpmap:98 iLBC/8000
               Media Attribute (a): rtpmap:101 telephone-event/8000
29             Media Attribute (a): sendrecv
```

Das Paket in Listing 29 wurde ebenfalls über die gesicherte Verbindung zwischen dem Server OBSD2 und OBSD1 versandt (Zeile 1, 2 und 3). Der `INVITE` Request soll an `birndl@ekiga.net` weitergereicht werden (Zeile 6). Als Contact Adresse dient `obsd2@192.168.2.2` (Zeile 10). Das Softphone hat via STUN die externe IP Adresse `213.150.1.160` herausgefunden. Diese wird im SDP Teil des Paketes eingetragen. Der SIP-Proxy sucht via DNS Abfrage den Inbound Proxy für die Domain `ekiga.net` und leitet den Request weiter:

Listing 30: INVITE Request Proxy -> öffentliches Netz

```
1  Internet Protocol, Src: 213.150.1.161 (213.150.1.161), Dst: 86.64.162.35 (86.64.162.35)
   User Datagram Protocol, Src Port: 63332 (63332), Dst Port: 5060 (5060)
3  Session Initiation Protocol
       Request-Line: INVITE sip:birndl@ekiga.net SIP/2.0
5      Message Header
           Record-Route: <sip:192.168.1.1;lr=on;ftag=0a320a7d>
7          Via: SIP/2.0/UDP 192.168.1.1;branch=z9hG4bK05f4.2b3991b1.0
           Via: SIP/2.0/UDP 192.168.2.2:13694;branch=z9hG4bK-d87543-275a9a5f121a365b-1--d87543-;rport=13694
9          Max-Forwards: 69
           Contact: <sip:obsd2@192.168.2.2:13694>
11         To: "birndl@ekiga.net"<sip:birndl@ekiga.net>
           From: "obsd2"<sip:obsd2@obsd1.stderror.at>;tag=0a320a7d
```

Dieses Paket ist natürlich **nicht** via ESP geschützt und kann daher leicht von Dritten abgehört werden. Sollte auch dieses Paket geschützt werden, müsste entweder eine weitere IPSec Verbindung bestehen, oder SIP über TLS (SIPS) verwendet werden (siehe RFC3261 [RSC+02]).

Nach Erhalt des `200 OK` Responses des User Agents im öffentlichen Netz beginnt der Media Stream zwischen den beiden UA's. Dieser verläuft aber direkt zwischen den beiden Endgeräten.

Listing 31: 200 OK SIP Proxy -> UA2

```
   Enc IPv4, SPI 0xd7d5bb2c, authentic, confidential
2  Internet Protocol, Src: 213.150.1.161 (213.150.1.161), Dst: 213.150.1.160 (213.150.1.160)
   Internet Protocol, Src: 192.168.1.1 (192.168.1.1), Dst: 192.168.2.2 (192.168.2.2)
4  User Datagram Protocol, Src Port: 5060 (5060), Dst Port: 13694 (13694)
   Session Initiation Protocol
6      Status-Line: SIP/2.0 200 OK
       Message Header
8          Via: SIP/2.0/UDP 192.168.2.2:13694;branch=z9hG4bK-d87543-275a9a5f121a365b-1--d87543-;rport=13694
           Record-Route: <sip:86.64.162.35;ftag=0a320a7d;lr=on>
10         Record-Route: <sip:192.168.1.1;lr;ftag=0a320a7d>
           Contact: <sip:birndl@213.150.1.156:38550;rinstance=4068e813d75cd2eb>
```

```
12          To: "birndl@ekiga.net"<sip:birndl@ekiga.net>;tag=317dab16
            From: "obsd2"<sip:obsd2@obsd1.stderror.at>;tag=0a320a7d
14          Call-ID: NzFjNzgxMDdiNmRhZGUzYzhhYjkyMzY5MWY5N2Y3ZjQ.
            CSeq: 1 INVITE
16          Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE, SUBSCRIBE, INFO
            Content-Type: application/sdp
18          User-Agent: X-Lite release 1006e stamp 34025
            Content-Length: 328
20      Message body
            Session Description Protocol
22              Session Description Protocol Version (v): 0
                Owner/Creator, Session Id (o): - 5 2 IN IP4 213.150.1.156
24              Session Name (s): CounterPath X-Lite 3.0
                Connection Information (c): IN IP4 213.150.1.156
26              Time Description, active time (t): 0 0
                Media Description, name and address (m): audio 59812 RTP/AVP 107 119 0 98 8 3 101
28              Media Attribute (a): alt:1 1 : Yl4G6P1+ sEgr9xfl 213.150.1.156 59812
                Media Attribute (a): fmtp:101 0-15
30              Media Attribute (a): rtpmap:107 BV32/16000
                Media Attribute (a): rtpmap:119 BV32-FEC/16000
32              Media Attribute (a): rtpmap:98 iLBC/8000
                Media Attribute (a): rtpmap:101 telephone-event/8000
34              Media Attribute (a): sendrecv
```

Abbildung 31 zeigt den `OK` Response. Es wurde die Contact Adresse `birndl@213.150.1.156` eingetragen (Zeile 11). Im SDP Teil findet sich ebenfalls diese öffentliche Adresse (Zeile 25).

Listing 32: Media Stream

```
Internet Protocol, Src: 192.168.2.2 (192.168.2.2), Dst: 213.150.1.156 (213.150.1.156)
2 User Datagram Protocol, Src Port: 18270 (18270), Dst Port: 59812 (59812)
Real-Time Transport Protocol
4
Internet Protocol, Src: 213.150.1.156 (213.150.1.156), Dst: 192.168.2.2 (192.168.2.2)
6 User Datagram Protocol, Src Port: 59812 (59812), Dst Port: 18270 (18270)
Real-Time Transport Protocol
```

In obiger Abbildung sind beispielhaft zwei Pakete, die zum Mediastream der Verbindung gehören, dargestellt. Man kann erkennen, dass die Verbindung direkt zwischen dem UA im internen LAN2 `192.168.2.2` und dem öffentlichen UA `213.150.1.156` verläuft. Die Pakete passieren zwar den Server OBSD2, dort wird aber nur die Source IP Adresse der Pakete von der internen Adresse `192.168.2.2` auf die externe Adresse `213.150.1.160` geändert.

4.2.4 Aufbau einer SIP Session Internet zu LAN2

In diesem Fall ruft ein UA im öffentlichen Netz den User Agent im internen LAN2 an. Wie bereits in Abschnitt 4.2.3 vorgenommen, folgt auf eine generelle Beschreibung im Abschnitt `Aufbau` eine genaue Analyse im Abschnitt `Ergebnisse`.

4.2.4.1 Aufbau

Der Verlauf der Signalisierung und des Mediastreams entspricht im wesentlichen Abschnitt 4.2.3.1

4.2.4.2 Ergebnisse

Bei der Registrierung gibt es keinen Unterschied zu Kapitel 4.2.1. Der UA im LAN2 sendet den `REGISTER` Request über die IPSec Verbindung an den SIP-Proxy. Dieser antwortet

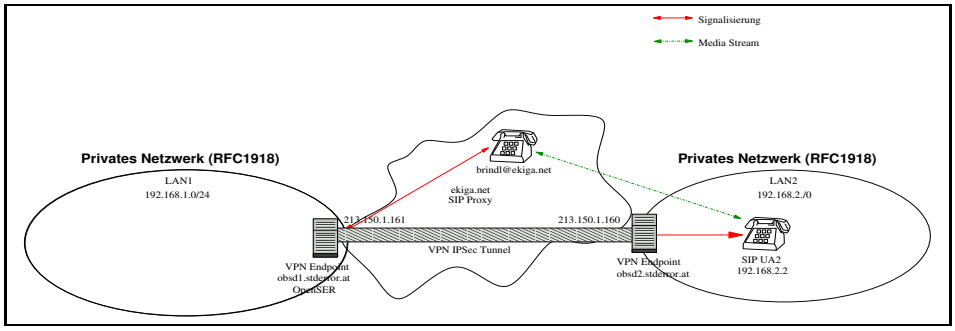

Abb. 12: Kommunikation LAN2 - öffentliches Netz

ebenfalls auf gleichem Weg.

Ruft der User Agent im öffentlichen Netz nun den User Agent im internen LAN2 an, erfolgt zuerst ein `INVITE` Request.

Listing 33: INVITE öffentlicher UA -> Proxy

```
1  Internet Protocol, Src: 213.150.1.156 (213.150.1.156), Dst: 213.150.1.161 (213.150.1.161)
   User Datagram Protocol, Src Port: 38550 (38550), Dst Port: 5060 (5060)
3  Session Initiation Protocol
       Request-Line: INVITE sip:obsd2@obsd1.stderror.at SIP/2.0
5      Message Header
           Via: SIP/2.0/UDP 213.150.1.156:38550;branch=z9hG4bK-d87543-166df410b3640f06-1--d87543-;rport
7          Max-Forwards: 70
           Contact: <sip:birndl@213.150.1.156:38550>
9          To: "obsd2@obsd1.stderror.at"<sip:obsd2@obsd1.stderror.at>
           From: "Thomas Mechtler"<sip:birndl@ekiga.net>;tag=2821764d
11         Call-ID: OWI2ZWI0ODY2NmYxNzFmZjQ5ODE3NTZjNTljODVlNjU.
           CSeq: 1 INVITE
13         Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE, SUBSCRIBE, INFO
           Content-Type: application/sdp
15         User-Agent: X-Lite release 1006e stamp 34025
           Content-Length: 277
17     Message body
           Session Description Protocol
19             Session Description Protocol Version (v): 0
               Owner/Creator, Session Id (o): - 8 2 IN IP4 213.150.1.156
21             Session Name (s): CounterPath X-Lite 3.0
               Connection Information (c): IN IP4 213.150.1.156
23             Time Description, active time (t): 0 0
               Media Description, name and address (m): audio 34406 RTP/AVP 107 119 0 98 8 3 101
25             Media Attribute (a): fmtp:101 0-15
               Media Attribute (a): rtpmap:107 BV32/16000
27             Media Attribute (a): rtpmap:119 BV32-FEC/16000
               Media Attribute (a): rtpmap:98 iLBC/8000
29             Media Attribute (a): rtpmap:101 telephone-event/8000
               Media Attribute (a): sendrecv
```

Der Request wird direkt vom UA mit der IP Adresse `213.150.1.156` an den SIP-Proxy am Server OBSD1 versandt (Zeile 1). In diesem Fall wird kein Outbound Proxy verwendet, sondern das Softphone am externen Client sucht selbstständig im DNS nach dem zuständigen Inbound Proxy. In `Via` Header (Zeile 6) findet sich daher nur die Adresse des Softphones. Es wird die Contact Adresse `birndl@213.150.1.156` (Zeile 8) verwendet. Im SDP Teil findet sich auch die IP Adresse des Telefons (Zeile 20). Der SIP-Proxy reicht das Paket über die VPN Verbindung an den Client in LAN2 weiter:

Listing 34: INVITE SIP Proxy -> UA2

```
   Enc IPv4, SPI 0xd7d5bb2c, authentic, confidential
2  Internet Protocol, Src: 213.150.1.161 (213.150.1.161), Dst: 213.150.1.160 (213.150.1.160)
   Internet Protocol, Src: 192.168.1.1 (192.168.1.1), Dst: 192.168.2.2 (192.168.2.2)
4  User Datagram Protocol, Src Port: 5060 (5060), Dst Port: 13694 (13694)
   Session Initiation Protocol
6      Request-Line: INVITE sip:obsd2@192.168.2.2:13694;rinstance=8f687f8744c81e83 SIP/2.0
```

Die Zeilen 1,2 und 3 in Listing 34 zeigen wieder, dass das Paket über die IPSec Verbindung übertragen wurde. Der `INVITE` Request wurde allerdings vom SIP-Proxy verändert (Zeile 6).

Es befindet sich nun die Kontaktadresse des UA im LAN2 `192.168.2.2` im Request.

Der UA antwortet ebenfalls über die VPN Verbindung:

Listing 35: 200 OK UA2 -> SIP-Proxy OBSD1

```
Enc IPv4, SPI 0xcf528764, authentic, confidential
Internet Protocol, Src: 213.150.1.160 (213.150.1.160), Dst: 213.150.1.161 (213.150.1.161)
Internet Protocol, Src: 192.168.2.2 (192.168.2.2), Dst: 192.168.1.1 (192.168.1.1)
User Datagram Protocol, Src Port: 13694 (13694), Dst Port: 5060 (5060)
Session Initiation Protocol
    Status-Line: SIP/2.0 200 OK
    Message Header
        Via: SIP/2.0/UDP 192.168.1.1;branch=z9hG4bK62e2.c13040f6.0
        Via: SIP/2.0/UDP 213.150.1.156:38550;branch=z9hG4bK-d87543-166df410b3640f06-1--d87543-;rport=38550
        Record-Route: <sip:192.168.1.1;r2=on;lr=on;ftag=2821764d>
        Record-Route: <sip:213.150.1.161;lr;r2=on;ftag=2821764d>
        Contact: <sip:obsd2@192.168.2.2:13694;rinstance=8f687f8744c81e83>
        To: "obsd2@obsd1.stderror.at"<sip:obsd2@obsd1.stderror.at>;tag=b64f2555
        From: "Thomas␣Mechtler"<sip:birndl@ekiga.net>;tag=2821764d
        Call-ID: OWI2ZWI0ODY2NmYxNzFmZjQ5ODE3NTZjNTljODVlNjU.
        CSeq: 1 INVITE
        Allow: INVITE, ACK, CANCEL, OPTIONS, BYE, REFER, NOTIFY, MESSAGE, SUBSCRIBE, INFO
        Content-Type: application/sdp
        User-Agent: X-Lite release 1006e stamp 34025
        Content-Length: 277
    Message body
        Session Description Protocol
            Session Description Protocol Version (v): 0
            Owner/Creator, Session Id (o): - 4 2 IN IP4 213.150.1.160
            Session Name (s): CounterPath X-Lite 3.0
            Connection Information (c): IN IP4 213.150.1.160
            Time Description, active time (t): 0 0
            Media Description, name and address (m): audio 50574 RTP/AVP 107 119 0 98 8 3 101
            Media Attribute (a): fmtp:101 0-15
            Media Attribute (a): rtpmap:107 BV32/16000
            Media Attribute (a): rtpmap:119 BV32-FEC/16000
            Media Attribute (a): rtpmap:98 iLBC/8000
            Media Attribute (a): rtpmap:101 telephone-event/8000
            Media Attribute (a): sendrecv
```

Listing 35 zeigt, dass der Response wieder über die VPN Verbindung übertragen wird (Zeile 1,2 und 3). Als Contact Adresse wird `obsd2@192.168.2.2` angegeben (Zeile 12). Im SDP Teil findet sich die via STUN erkannte öffentliche IP Adresse des UA (Zeile 26). Der SIP Proxy leitet den Request direkt an das Telefon weiter:

Listing 36: 200 OK SIP Proxy -> öffentlicher UA

```
Internet Protocol, Src: 213.150.1.161 (213.150.1.161), Dst: 213.150.1.156 (213.150.1.156)
User Datagram Protocol, Src Port: 5060 (5060), Dst Port: 38550 (38550)
Session Initiation Protocol
    Status-Line: SIP/2.0 200 OK
    Message Header
        Via: SIP/2.0/UDP 213.150.1.156:38550;branch=z9hG4bK-d87543-166df410b3640f06-1--d87543-;rport=38550
        Record-Route: <sip:192.168.1.1;r2=on;lr=on;ftag=2821764d>
        Record-Route: <sip:213.150.1.161;lr;r2=on;ftag=2821764d>
        Contact: <sip:obsd2@192.168.2.2:13694;rinstance=8f687f8744c81e83>
        To: "obsd2@obsd1.stderror.at"<sip:obsd2@obsd1.stderror.at>;tag=b64f2555
        From: "Thomas␣Mechtler"<sip:birndl@ekiga.net>;tag=2821764d
        Call-ID: OWI2ZWI0ODY2NmYxNzFmZjQ5ODE3NTZjNTljODVlNjU.
        CSeq: 1 INVITE
```

In Listing 36 erkennt man, dass das Paket direkt an den öffentlichen UA weitergegeben wird. Diese Übertragung erfolgt aber direkt und **nicht** über die mit IPSec abgesicherte Verbindung. Sollen auch diese Pakete geschützt werden, müsste ein zusätzlicher IPSec Tunnel aufgebaut werden oder es müsste TLS zum Einsatz kommen.

Der Media Stream zwischen den beiden Endgeräten verläuft wieder direkt:

Listing 37: Media Stream

```
Internet Protocol, Src: 192.168.2.2 (192.168.2.2), Dst: 213.150.1.156 (213.150.1.156)
User Datagram Protocol, Src Port: 50574 (50574), Dst Port: 34406 (34406)
Real-Time Transport Protocol

Internet Protocol, Src: 213.150.1.156 (213.150.1.156), Dst: 192.168.2.2 (192.168.2.2)
User Datagram Protocol, Src Port: 34406 (34406), Dst Port: 50574 (50574)
Real-Time Transport Protocol
```

Man erkennt in Listing 37, dass nur die beiden Kommunikationpartner `192.168.2.2` und `213.150.1.156` im Spiel sind. Die Pakete durchlaufen zwar den Server OBSD2, werden dort aber nur via NAT umgeschrieben.

5 Diskussion

Bereits im Grundlagenteil dieser Arbeit stellt sich heraus, dass mit IPSec eine sehr flexible, aber komplexe Sicherheitsarchitektur entstanden ist. OpenBSD vereinfacht zwar die Konfiguration von IPSec Verbindungen, ohne ausreichende Grundlagen kann aber weder das Funktionieren, noch die Sicherheit garantiert werden.

Der Test einer LAN zu LAN Verbindung lief ohne große Überraschung ab. Sowohl die Signalisierungsdaten als auch der Media Stream werden durch den IPSec Tunnel abgesichert. Eine Änderung brachte erst die Aktivierung des STUN Protokolls. Da den Telefonen bei aktiviertem STUN auch die externen IP Adressen bekannt sind, ist ein Verlauf der Verbindungsdaten nicht mehr so eindeutig vorhersagbar. Dies verdeutlicht schon der SIP `Register` Request, da sich die Telefone im LAN nun sowohl mit ihrer LAN IP-Adresse registrieren, als auch mit ihrer durch STUN ermittelten externen Adresse. Es muss genauestens darauf geachtet werden, welche Wege die Signalisierung bzw. der Media Stream zwischen den beiden Telefonen nimmt. Da im SDP Teil des SIP Verbindungsaufbaus die externen Adressen als Kontaktadressen für eine Mediaverbindung eingetragen wurden, erwartete der Autor eine Kommunikation, die über das öffentliche Netzwerk abläuft. Die Softphones handeln hier aber autonom und erkennen, dass eine Verbindung auch mit Hilfe der internen - durch das VPN gesicherten - Verbindung möglich ist. Überraschender Weise verläuft also der Media Stream, trotz öffentlicher Adressen im SDP Teil, über die geschützte Verbindung. Hier stellt sich das Problem, dass dies eine spezifische Eigenschaft der verwendeten Softphones ist. Eine neue Version der Software bzw. ein Softphone eines anderen Hersteller, kann hier schon Änderungen im Verbindungsablauf bedeuten. Gleiches gilt wahrscheinlich auch bei der Verwendung von Hardware Voice over IP Telefonen (z.B. Snom 300).

Der Verbindungsaufbau LAN2 zu Internet bzw. Internet zu LAN2 brachte die Erkenntnis, dass hier der Signalisierungsverkehr zwar über die gesicherte VPN Verbindung von Statten geht, der Media Stream aber ungesichert verläuft. Dies ist zwar auf Grund der verwendeten Netzwerkarchitektur nicht anders möglich, scheint aber trotzdem erwähnenswert. Sollte auch der Mediastream kryptografisch abgesichert werden, müsste entweder eine weitere IPSec Verbindung definiert werden oder auf das Secure Realtime Protocol (SRTP) zurückgegriffen werden.

Der Autor ist auch der Meinung, dass auf Grund dieser Komplexität IPSec für eine End-to-End Absicherung der Verbindung zwischen User Agents nicht geeignet ist. Einem Nicht-Informatiker ist die Konfiguration wohl nicht zuzutrauen. Für die Verbindung von verschiedenen privaten Netzwerken über ein öffentliches Netzwerk ist IPSec aber gut geeignet. Es verwendet moderne Verschlüsselungsalgorithmen und hat sich als sehr stabil erwiesen. Hat man die Hürde der Grundlagen und der Erstkonfiguration hinter sich, sind keine großen Probleme mehr zu erwarten. Allerdings sollte ein besonderes Augenmerk auf das Monitoring der Performance der Verbindung gelegt werden. Verschlüsselung bedeutet einen erhöhten Verbrauch von CPU Ressourcen und somit ist der Anzahl der möglichen Verbindungen eine Grenze gesetzt. Bei einigen wenigen Telefonaten wird sich dies noch nicht bemerkbar machen, bei tausenden Verbindungen können hier jedoch Probleme auftreten. Dies sollte auch bei der Auswahl eines geeigneten IPSec Gateways berücksichtigt werden. OpenBSD nutzt selbst bei Multiprozessor Architekturen nur eine CPU und könnte somit zum Flaschenhals werden.

Quality of Service stellt ebenfalls ein Problem bei geschützten IPSec Verbindungen dar. Sollen Netzwerkkomponenten Rücksicht auf Voice over IP Verbindungen nehmen und diesen eine gewisse Bandbreite im Netzwerk garantieren, muss QoS verwendet werden. Dies ist aber bei verschlüsselten Paketen nicht möglich, da Flags, die im IP Header gesetzt sind, durch eine Verschlüsselung für Router nicht mehr erkennbar sind.

Einer eventuellen Verzögerung beim erstmaligen Verbindungsaufbau über die gesicherte VPN Verbindung sollte ebenfalls Augenmerk geschenkt werden. Bei der Verwendung von OpenBSD tratten diese Probleme zwar nicht auf, dies ist aber von der IPSec Implementierung abhängig. Andere IPSec Stacks können sich hier anders verhalten, z.B. ist dem Autor auch die Implementierung des FreeBSD IPSec Stacks bekannt. Dort treten beim erstmaligen Versenden von Paketen über eine VPN Verbindung Verzögerungen von einigen Sekunden auf. Dies kann natürlich auch Einfluss auf eine Voice over IP Verbindung nehmen.

Weiters können noch Inkompatibilitäten zwischen verschiedenen IPSec Gateways auftreten. IPsec, ISAKMP und IKE sind zwar durch RFC's spezifiziert, dies hindert aber Hersteller nicht daran, Erweiterungen bzw. Änderungen vorzunehmen. Vor eine Implementierung sollte daher untersucht werden, ob es zu Problemen zwischen verschiedenen IPSec Stacks kommt. Am einfachsten ist es hier allerdings, sich auf eine Implementierung zu konzentrieren.

Der Authentifizierung zwischen den IPSec Gateway muss ebenfalls Beachtung geschenkt werden. In dieser Arbeit wurde der Einfachheit halber ein Public Key Verfahren gewählt. Dies ist bei wenigen Gateway's auch völlig ausreichend. Sollen allerdings eine Vielzahl von Netzwerken miteinander verbunden werden, sind der Skalierung dieses Verfahrens Grenzen gesetzt. Bei der in dieser Arbeit verwendeten Methode müssen allen Gateways, die an einem Verbund teilnehmen, auch alle öffentlichen Schlüssel der Verbindungspartner bekannt sein. Sollen eine Vielzahl von Netzwerken verbunden werden, ist es eine Authentifizierung auf Zertifikatsbasis anzuraten. Jedes IPSec Gateway bekommt ein Zertifikat ausgestellt und nimmt Verbindungen nur dann an, falls sich eine Gegenstelle meldet, die ein Zertifikat besitzt, das von der gleichen Zertifizierungsautorität ausgestellt wurde. Die Einführung einer notwendigen Publick Key Infrastructure (PKI) ist aber nicht trivial und sollte sorgfältig geplant werden. Vor allem das Zurückziehen von bereits ausgestellten Zertifikaten stellt meistens ein Problem dar. Einer Verwendung von Pre-Shared Keys ist abzuraten, da hier die Sicherheit alleinig von der Qualität des Passwortes abhängig ist. Wird dieses Passwort kompromitiert, ist die Sicherheit des gesamten Netzwerkverbundes nicht mehr gewährleistet.

Durch die Vermischung der OSI-Layer im SIP und SDP Protokoll ist es erst durch weitreichende Analysen ersichtlich, wie welcher Verbindungsablauf von Statten geht. Es muss genauestens untersucht werden, welche Wege im Netzwerk gewählt werden. Die Intelligenz der Telefone stellt hier eine weitere Herausforderung dar. Es ist anzuraten, vor einem Einsatz eines bestimmten Telefons eine genaue Untersuchung der Verbindungseigenschaften des Telefons vorzunehmen. Erst dadurch kann vermieden werden, dass eventuell erst zu spät erkannt wird, dass Teile einer Verbindung ungesichert verlaufen.

Abbildungsverzeichnis

Tabellenverzeichnis

Listings

Abkürzungsverzeichnis

AES ..... Advanced Encryption Standard
DNS ..... Domain Name Service
DOI ..... Domain of Interpretation
DOS ..... Denial of Service
DSS ..... Digital Signature Standard
ICV ..... Integrity Check Value
IKE ..... Internet Key Exchange Prfotokoll
IPSec ..... IP Security Protokoll
ISAKMP ..... Internet Security Association and Key Managment Protokoll
IV ..... Initialization Vector
NAT ..... Network Address Translation
OSPF ..... Open Shortest Path First
PKI ..... Public Key Infrastructure
QoS ..... Quality of Service
SA ..... Security Association
SAD ..... Security Association Database
SDP ..... Session Description Protokoll
SIP ..... Session Initiation Protokoll
SIPS ..... SIP over TLS
SNMP ..... Simple Network Managment Protokoll
SPI ..... Security Parameter Index
SRTP ..... Secure Realtime Protocol
TLS ..... Transaction Layer Security
UA ..... User Agent
UAC ..... User Agent Client
UAS ..... User Agent Server

Literaturverzeichnis

Literatur

[HC98] D. Harkins and D. Carrel. The Internet Key Exchange (IKE). RFC 2409 (Proposed Standard), November 1998. Obsoleted by RFC 4306, updated by RFC 4109.

[HJP06] M. Handley, V. Jacobson, and C. Perkins. SDP: Session Description Protocol. RFC 4566 (Proposed Standard), July 2006.

[Kau05] C. Kaufman. Internet Key Exchange (IKEv2) Protocol. RFC 4306 (Proposed Standard), December 2005.

[Ken05a] S. Kent. IP Authentication Header. RFC 4302 (Proposed Standard), December 2005.

[Ken05b] S. Kent. IP Encapsulating Security Payload (ESP). RFC 4303 (Proposed Standard), December 2005.

[Kra96] H. Krawczky. Skeme: A verastile secure key exchange mechanism for internet, 1996.

[KS05] S. Kent and K. Seo. Security Architecture for the Internet Protocol. RFC 4301 (Proposed Standard), December 2005.

[Lin06] Alexander Lintenhofer. Zusammenspiel zwischen ipsec sa, sad und spd, 2006.

[MSST98] D. Maughan, M. Schertler, M. Schneider, and J. Turner. Internet Security Association and Key Management Protocol (ISAKMP). RFC 2408 (Proposed Standard), November 1998. Obsoleted by RFC 4306.

[Orm98] H. Orman. The OAKLEY Key Determination Protocol. RFC 2412 (Informational), November 1998.

[Pip98] D. Piper. The Internet IP Security Domain of Interpretation for ISAKMP. RFC 2407 (Proposed Standard), November 1998. Obsoleted by RFC 4306.

[RMK+96] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, and E. Lear. Address Allocation for Private Internets. RFC 1918 (Best Current Practice), February 1996.

[RS02] J. Rosenberg and H. Schulzrinne. Session Initiation Protocol (SIP): Locating SIP Servers. RFC 3263 (Proposed Standard), June 2002.

[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 (Proposed Standard), June 2002. Updated by RFCs 3265, 3853, 4320.

[RWHM03] J. Rosenberg, J. Weinberger, C. Huitema, and R. Mahy. STUN - Simple Traversal of User Datagram Protocol (UDP) Through Network Address Translators (NATs). RFC 3489 (Proposed Standard), March 2003.

[SE01] P. Srisuresh and K. Egevang. Traditional IP Network Address Translator (Traditional NAT). RFC 3022 (Informational), January 2001.

6 Anhang A

Listing 38: OpenSER 1.1.0 patch

```
--- sip-server.orig/modules/postgres/aug_sysdep.h       Wed Jul  5 12:55:08 2006
+++ sip-server/modules/postgres/aug_sysdep.h    Mon Dec  4 11:00:43 2006
@@ -309,10 +309,48 @@

 /*
 ----------------------------------------------------------------------
+----- OpenBSD x86 with GCC ---------------------------------------------
+----------------------------------------------------------------------
 */
+#ifdef __OpenBSD__

+#define AUG_HAS_LPR
+#define AUG_HAS_RANDOM
+#define AUG_HAS_PSAX
+#define AUG_HAS_PREAD
+#define AUG_NO_CRYPT_H
+#define AUG_NO_TERMIO_H
+#define AUG_NO_DB
+
+/*  OpenBSD lacks these error codes.  */
+#define ENODATA         ENOBUFS
+#define EUNATCH         ENOPROTOOPT
+#ifndef EPROTO
+#define EPROTO EPROTOTYPE
+#endif
+
+/*  OpenBSD lacks these termios codes.  */
+#define TCGETS TIOCGETA
+#define TCSETS TIOCSETA
+// #define TCGETA       TIOCGETA
+// #define TCSETA       TIOCSETA
+#define TCSETSW         TIOCSETAW
+// #define TCFLSH       TIOCFLUSH
+#define termio termios
+
+#define AUG_CONFIGURATION       "OpenBSD"
+
+typedef unsigned int augUInt32;
+
+#endif /* openbsd */
+
+/*
+----------------------------------------------------------------------
+*/
+
 #ifndef AUG_CONFIGURATION
 #error "os/cpu/compiler combination not configured in $Source: /home/pinhead/CVSREPO/fh/5/tks/bsc/sections/anhang.tex,v

 #endif /* AUG_SYSDEP_H */
+
```