

# 3 VPN Protokolle

Es existieren mehrere verschiedene Protokolle, die für den Aufbau eines VPNs genutzt werden können. Dieses Buch nutzt in erster Linie die IPsec Protokolle. Daher beschäftigt sich dieses Kapitel auch vorrangig mit diesen Protokollen. Da jedoch auch das L2TP Protokoll immer stärker in Kombination mit IPsec genutzt wird, wird auch dieses Protokoll kurz vorgestellt. Das PPTP Protokoll wird heute nur noch selten eingesetzt. Eine Unterstützung unter Linux ist möglich, aber nicht Thema dieses Buches.

## 3.1 Einleitung

Der Bedarf, Daten verschlüsselt über Computernetze zu übertragen, ist so alt wie die Computernetze selbst. Die eigentlichen Netzwerkprotokolle, wie zum Beispiel das Internet Protokoll IP, bieten diese Funktion jedoch nicht. Daher gibt es eine Vielzahl von verschiedenen Protokollen, die dies basierend auf den Netzwerkprotokollen ermöglichen. Einige dieser Protokolle haben einen höheren Bekanntheitsgrad erreicht (zum Beispiel das Secure Shell Protokoll und die Secure Socket Layer (SSL)) und wurden standardisiert (zum Beispiel Transaction Layer Security (TLS)). Meist sind die Protokolle für eine ganz bestimmte Anwendung entwickelt worden. So wurde das SSL Protokoll ursprünglich für den Schutz der HTTP-Kommunikation zwischen einem Browser und einem Webserver entworfen.

Die Protokolle, die in einem VPN eingesetzt werden, unterscheiden sich in ihrer Universalität meist von den anwendungsorientierten Protokollen. Sie ermöglichen es sämtliche Daten und alle Kommunikationsströme unabhängig vom verwendeten Protokoll gemeinsam zu verschlüsseln und ihre Vertraulichkeit und Integrität zu garantieren.

Üblicherweise werden dazu die zu übertragenden Nutzpakete komplett im VPN Protokoll abgekapselt. Um dies zu unterstützen muss das VPN-Protokoll in der Lage sein zum Beispiel IP-Pakete zu übertragen. Das SSL-Protokoll kann lediglich die Pakete einer beliebigen TCP Verbindung übertragen. SSH ist nur in der Lage ASCII Verbindungen zu ermöglichen und einzelne TCP Verbindungen zu tunneln.

Erste Versuche des Aufbaus eines VPNs unter UNIX oder Linux basierten häufig auf der Secure Shell. Hierzu wird eine Secure-Shell-Verbindung zwischen zwei UNIX Systemen aufgebaut. Über diese Verbindung können nun

ASCII Daten ausgetauscht werden. Anschließend wird auf beiden Seiten der Point-to-Point-Protokoll-Daemon (PPP-Daemon) gestartet. Der PPP-Daemon kommuniziert nun über die Secure-Shell-Verbindung und überträgt Netzwerkpakete. Beim Start des PPP-Daemon wurde automatisch auf beiden Systemen eine zusätzliche Netzwerkkarte `ppp0` initialisiert. Alle Pakete, die an diese Karte geschickt werden, werden nun vom PPP-Daemon über die Secure-Shell-Verbindung an den zweiten Rechner geschickt, dort entgegen genommen und über die `ppp0` Karte wieder zur Verfügung gestellt. Existieren nun entsprechende Routingeinträge auf den Rechnern, so kann diese Verbindung als einfaches VPN genutzt werden. Das Linux VPN HowTo (<http://www.tldp.org/HOWTO/VPN-HOWTO/>) beschreibt einen derartigen Aufbau.

Das erste in großem Stil eingesetzte VPN Protokoll war das Point-to-Point-Tunneling-Protokoll (PPTP). Dieses Protokoll wurde unter anderem von Microsoft entwickelt und steht in vielen Microsoft Windows Betriebssystemen (Windows 9x, ME, NT, 2000) zur Verfügung.

Bei genauer Betrachtung ähnelt das PPTP Protokoll der Variante mit SSH/PPP. Das PPTP Protokoll ermöglicht den Aufbau eines GRE (Generic Routing Encapsulation) Tunnels zwischen zwei Rechnern. Auf der Basis dieses GRE Tunnels wird auf beiden Seiten der PPP Daemon gestartet. Der PPP Daemon übernimmt die IP Pakete und verpackt sie in PPP Pakete, die über den GRE Tunnel transportiert werden. Das PPTP Protokoll ist aber im Gegensatz zum SSH Protokoll beim oben beschriebenen Tunnel in keiner Weise für die Verschlüsselung oder Authentifizierung zuständig. Es überträgt lediglich die für die PPP Sitzung zu verwendenden IP Adressen. Die Authentifizierung wird vom PPP Daemon durchgeführt. Hierbei erfolgt die Anmeldung mit einem Benutzernamen und einem Kennwort.

Um dieses Protokoll dennoch als Basis für ein VPN einsetzen zu können wurde von Microsoft das PPP-Protokoll erweitert. Für eine sichere Authentifizierung wurde von Microsoft ein Challenge/Response Authentication Protokoll (CHAP) in Form von MS-CHAP implementiert. Das wurde, nachdem einige Fehler bekannt wurden, als MS-CHAPv2 überarbeitet vorgestellt. Zusätzlich soll ein VPN die Vertraulichkeit der übertragenen Pakete sicherstellen. Hierzu wurde von Microsoft der PPP-Daemon um das Microsoft-Point-to-Point-Encryption-Protokoll (MPPE) erweitert. Dieses Protokoll erlaubt die Verschlüsselung der Pakete mit 40 oder 128 Bit. Der hierbei verwendete Schlüssel wird vom eingesetzten Kennwort abgeleitet und typischerweise alle 256 Pakete neu ausgetauscht.

Die wesentlichen Schwäche des PPTP-Protokolls liegt in der Authentifizierung, aber es wurden auch weitere verschiedene Schwächen im MS-CHAP

Protokoll nachgewiesen. Bruce Schneier hat diese in mehreren Dokumenten aufgeführt (<http://www.counterpane.com/pptp.html>). Das größte Problem stellt die Authentifizierung mit einem Kennwort dar. Kennworte werden üblicherweise nicht zufällig erzeugt und sind daher von geringer Entropie. Es ist wesentlich einfacher ein Kennwort zu raten, als einen Brute Force-Angriff auf einen 128 Bit langen Schlüssel durchzuführen.

Heute wird das PPTP Protokoll nur noch selten für die Implementierung einer neuen VPN Lösung eingesetzt. Selbst Microsoft hat erkannt, dass das PPTP Protokoll kein Vertrauen mehr genießt und einen freien IPsec/L2TP Client für die Betriebssysteme veröffentlicht, die bisher keine IPsec Unterstützung bieten (<http://www.microsoft.com/windows2000/server/evaluation/news/bulletins/l2tpclient.asp>).

## 3.2 IPsec

Bei der Entwicklung des Protokolls IP Version 4 wurden Sicherheitsaspekte vernachlässigt. Die Betonung wurde auf Geschwindigkeit und Robustheit der Anwendung gelegt. Bei der Entwicklung des IP-Protokolls Version 6 sollten derartige Fehler von Beginn an vermieden werden. Dies führte zur Entwicklung der IPsec Protokolle, die im Anschluss auf das IP Protokoll Version 4 portiert wurden.

Die IPsec-Protokolle können die Vertraulichkeit, Authentizität und Integrität der übertragenen Daten garantieren. Hierfür stehen das Authentication Header-Protokoll (AH) und das Encapsulated-Security-Payload-Protokoll (ESP) zur Verfügung. Diese Protokolle bieten zusätzlich einen Schutz vor Replay Angriffen. Dabei werden mit einem Schiebefenster automatisch wiederholt gesendete Pakete erkannt und verworfen.

Für die Verschlüsselung, Authentifizierung und die Integritätsüberprüfung werden symmetrische Schlüssel benötigt. Sie müssen vorher ausgehandelt werden. Um dies automatisch zu ermöglichen wurde das Internet Key Exchange-Protokoll (IKE) entwickelt. IKE authentifiziert die Kommunikationspartner mit geeigneten Mitteln und verhandelt die zu verwendenden Algorithmen und Verwaltungsinformationen. Hierzu erzeugt das IKE Protokoll auch mit dem Diffie Hellmann-Verfahren symmetrische Schlüssel in der benötigten Anzahl.

Bei der Entwicklung des IPsec Protokolls wurden viele Aspekte des IPv4 Protokolls vernachlässigt. Dies ist verständlich, da die IPsec Protokolle zunächst für IPv6 entwickelt wurden. IPv6 wird wahrscheinlich keine Network Address Translation (NAT) benötigen, da der Adressraum mit 128 Bit langen

Netzwerkadressen wesentlich größer ist als der IPv4 Adressraum mit 32 Bit langen Netzwerkadressen. Die IPsec Protokolle unterstützen daher kein NAT. Neue Erweiterungen der IPsec Protokolle erlauben es diese erneut in UDP Paketen zu kapseln. Da das UDP Protokoll ein NAT unterstützt, erben die IPsec Protokolle diese Funktion.

Eine weitere Erweiterung der IPsec Protokolle stellt die DHCP-over-IPsec Funktion dar. Hierbei besteht die Möglichkeit über den IPsec Tunnel IP Adressen auszutauschen, die anschließend für die Verbindung genutzt werden.

Beide Protokollerweiterungen werden am Ende dieses Kapitels vorgestellt.

### 3.2.1 Integrität und Authentifizierung

Die IPsec Protokolle stellen die Integrität der übertragenen Pakete sicher und authentifizieren die Quelle der Pakete. Hierzu werden Hash-Algorithmen (siehe Abschnitt 2.6, »Hash-Funktion«) im Hash-Message-Authentication-Code (RFC 2104) Verfahren eingesetzt. Dabei wird der Hash-Wert aus einem geheimen, nur den Kommunikationspartnern bekannten, Schlüssel und den zu schützenden Daten ermittelt. Das Ergebnis, der HMAC, stellt gewissermaßen eine Signatur dar und erlaubt es, die Herkunft und die Integrität des Paketes zu testen.

Die IPsec Protokolle verwenden einen HMAC mit einer Länge von 96 Bits. Dieser HMAC kann mit den verschiedensten Protokollen ermittelt werden. Die IPsec Standards verlangen die Implementierung von HMAC-MD5-96 (RFC 2401) und HMAC-SHA-1-96 (RFC 2404). Hierbei wird bei MD5 ein geheimer Schlüssel von 128 Bit und bei SHA-1 ein geheimer Schlüssel von 160 Bit eingesetzt. Aus diesem Schlüssel und den zu schützenden Daten berechnet der Algorithmus einen HMAC. Dieser HMAC ist bei MD5 128 Bit lang und bei SHA-1 160 Bit lang. Die höchstwertigen 96 Bit dieses Schlüssels werden tatsächlich als HMAC (auch ICV Integrity Check Value) von den IPsec Protokollen verwendet (siehe Abbildung 3.1).

Der geheime Schlüssel und der eingesetzte HMAC Algorithmus werden entweder vom Administrator von Hand festgelegt (manuell verschlüsselte Verbindungen) oder das IKE Protokoll ermittelt diese Parameter und erzeugt die entsprechenden Security Associations.

Seit einigen Jahren werden zusätzliche HMAC Varianten entwickelt. So existieren Drafts, die die Verwendung von SHA-2 mit 256, 384 und 512 Bit Länge beschreiben. Weitere teilweise für IPsec weniger gebräuchliche HMAC Verfahren sind HMAC-RIPEMD-160, HMAC-PANAMA und HMAC-TIGER.

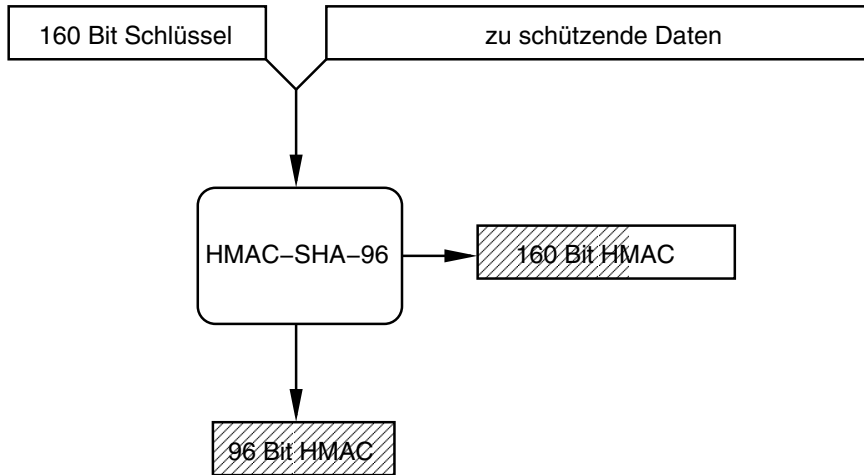


Abbildung 3.1 Der HMAC wird aus einem geheimen Schlüssel und den zu sichernden Daten ermittelt. Verwendet werden die 96 höchstwertigen Bits.

### 3.2.2 Verschlüsselung

Die IPsec Protokolle stellen die Vertraulichkeit der übertragenen Daten durch eine symmetrische Verschlüsselung der Informationen sicher. Hierbei werden die Daten mit einem geheimen Schlüssel im Cipher-Block-Chaining Modus mit symmetrischen Verfahren verschlüsselt. Lediglich die Kommunikationspartner verfügen über den geheimen Schlüssel und können so die Vertraulichkeit der Daten garantieren.

Die IPsec Protokolle können unterschiedliche Verschlüsselungsverfahren mit unterschiedlichen Schlüssellängen einsetzen. Die IPsec Standards verlangen die Unterstützung von NULL (RFC 2410), CBC-DES (RFC 1829) und CBC-DES mit expliziten IV (RFC 2405).

Die DES Verschlüsselung verwendet jedoch nur einen 64 Bit langen Schlüssel. 8 Bits dieses Schlüssels dienen lediglich als Paritätsinformation. Daher beträgt die wirksame Länge des Schlüssels nur 56 Bit. Aus diesem Grund werden heute weitere CBC-Verschlüsselungsverfahren von den meisten IPsec Implementierungen unterstützt (RFC 2451): CAST-128, RC5, IDEA, Blowfish und 3DES. Zusätzliche Drafts beschreiben die Verwendung von AES oder RC6.

Sämtliche eingesetzten Verschlüsselungsverfahren sind Block-Ciphern. Das bedeutet, dass diese die zu schützenden Daten in ganzen Blöcken verarbeiten. Die Blocklänge variiert und hängt vom Verfahren ab. Da in den seltens-

ten Fällen die zu verschlüsselnden Daten exakt ein Vielfaches der Blocklänge sind, ist ein Auffüllen der Daten bis zur nächsten Blockgrenze erforderlich. Dies wird als Padding bezeichnet.

Da es sich in allen Fällen um monoalphabetische Verschlüsselungsverfahren (siehe Abschnitt 2.4, »Cipher Block Chaining (CBC)«) handelt, ist es erforderlich, das Cipher Block Chaining-Verfahren einzusetzen, um einen Angriff durch eine Frequenzanalyse zu verhindern. Dabei wird ein Initialisierungsvektor verwendet, der vor jeder Verschlüsselung mit dem Klartextblock exklusiv verknüpft wird. Der erste Initialisierungsvektor wird zunächst zufällig ermittelt. Anschließend wird ein verschlüsselter Block als Vektor des nächsten Blockes genutzt. Jedes verschlüsselte IPsec Paket enthält einen Initialisierungsvektor. Um den Zufallszahlengenerator nicht übermäßig zu belasten, wird meist auch bei allen folgenden Paketen als Initialisierungsvektor der letzte verschlüsselte Block des vorhergehenden Paketes verwendet. Getreu dem Motto »Ein Bild sagt mehr als tausend Worte.« gibt die Abbildung 3.2 dies graphisch wieder.

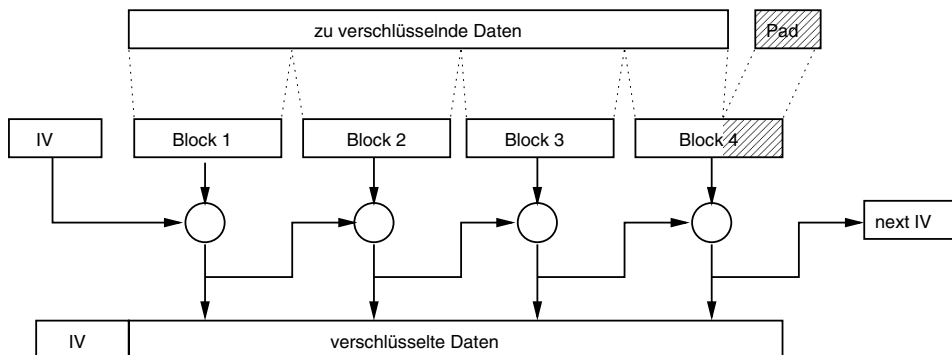


Abbildung 3.2 Im CBC Modus werden die zu verschlüsselnden Blöcke miteinander verknüpft. Der erste Block verwendet hierzu einen expliziten Initialisierungsvektor. Vor der Verschlüsselung ist unter Umständen ein Padding erforderlich.

### 3.2.3 Anti Replay Schutz

Die IPsec Protokolle implementieren einen optionalen Anti Replay Service. Hierzu muss der Absender jedes Paket mit einer (monoton) steigenden Sequenznummer versehen. Der Empfänger kann diese Nummer zum Schutz vor Replay Angriffen nutzen. Hierzu verwendet der Empfänger ein Schiebefenster bestimmter Größe (typisch sind Größen von 32 oder 64). Erhält der Empfänger nun ein Paket, dessen Sequenznummer links außerhalb des Fensters liegt, so wird dieses Paket sofort verworfen. Befindet sich die Sequenz-

nummer im Fenster und wurde das Paket bereits erhalten, so wird es ebenfalls verworfen. Lediglich neue Pakete im oder rechts vom Fenster werden nach erfolgreicher Authentifizierung akzeptiert und entschlüsselt. Befindet sich die Sequenznummer des akzeptierten Paketes rechts außerhalb des Fensters, so wird nach der erfolgreichen Authentifizierung das Fenster soweit nach rechts verschoben, dass sich dieses Paket nun auch innerhalb des Fensters befindet (siehe Abbildung 3.3).

Ein Angreifer, der die korrekten Pakete einer VPN Verbindung aufzeichnet und sie zu einem späteren Zeitpunkt wieder abspielt um einen Denial-of-Service zu erreichen, kann so erfolgreich daran gehindert werden. Würden nicht diese Sequenznummern eingesetzt werden, so müsste der Empfänger alle Pakete entschlüsseln, da er sie erfolgreich authentifizieren würde. Sie stammen ja alle vom korrekten Absender und wurden vom Angreifer nicht verändert. Nun werden veraltete und bereits erhaltene Pakete vor dem Test der Authentifizierung bereits verworfen. Die aufwändige Prüfung der Integrität und die Entschlüsselung des Paketes braucht nicht durchgeführt zu werden.

Das Fenster sollte nicht zu klein gewählt werden. Die Pakete können auf Grund der Struktur des Internets in unterschiedlicher Reihenfolge beim Empfänger eintreffen. Wird das Fenster zu klein gewählt, so sind häufige Neuübertragungen mit neuen Sequenznummern erforderlich. Die Mindestgröße beträgt laut Standard 32 Bit.

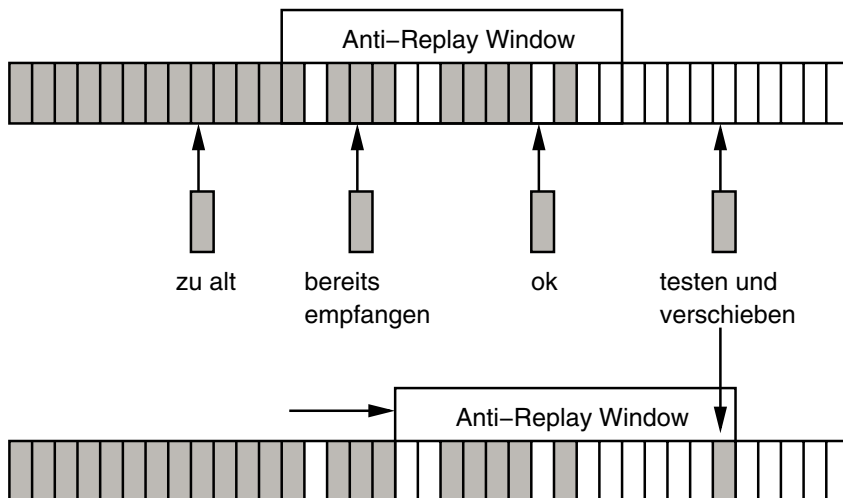


Abbildung 3.3 Das Anti-Replay Fenster schützt vor Replayangriffen mit alten aufgezeichneten Paketen

### 3.2.4 Tunnel- und Transport Modus

Die IPsec Protokolle unterstützen zwei verschiedene Modi zur Übertragung der Informationen: Tunnel Modus und Transport Modus. Im Transport Modus wird lediglich das Upper-Layer Protokoll (zum Beispiel TCP oder UDP) durch das IPsec Protokoll geschützt. Dabei wird der IPsec-Header zwischen dem IP-Header und dem Header des höheren Protokolls eingeschoben. Das *Next-Header* Feld im IPsec-Header enthält dann die Nummer des höheren Protokolls. Dieser Modus kann jedoch nur zum Einsatz kommen, wenn die beiden kommunizierenden Rechner direkt die Pakete verschlüsseln.

Häufiger wird daher der Tunnel Modus eingesetzt, bei dem die Rechner als VPN-Gateway fungieren und komplette IP-Pakete einschließlich ihrer Header schützen. Dabei wird das gesamte IP-Paket in einem IPsec-Header eingefasst, mit einem neuen IP-Header versehen und zum gegenüberliegenden Endpunkt des IPsec-Tunnels transportiert, wo es ausgepackt und mit seinem originalen IP-Header weitergeschickt wird. Das *Next-Header* Feld des IPsec-Headers enthält hier die Zahl 4 für das Protokoll IPv4 (siehe `/etc/protocols`).

Die Konstruktion der Header ist auch in der Abbildung 3.4 schematisch nochmals dargestellt.

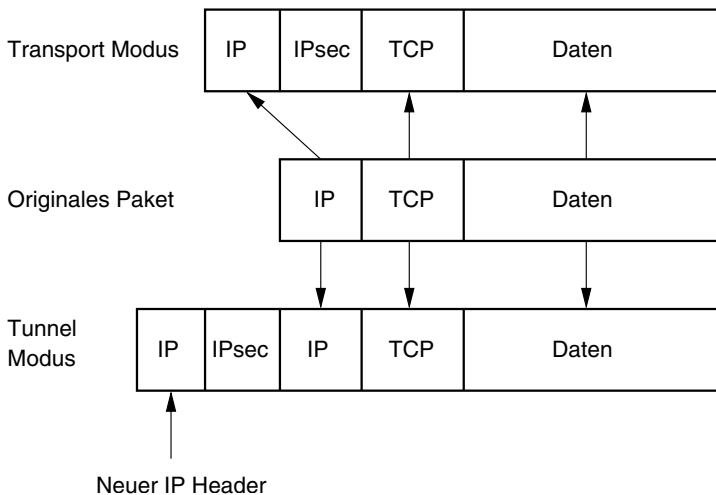


Abbildung 3.4 Im Tunnel Modus wird das gesamte IP-Paket durch einen IPsec-Header und einen neuen IP-Header gekapselt

### 3.2.5 Authentication Header – AH

Das Authentication Header Protokoll (IP Protokoll 51) stellt die Authentifizierung und die Integrität der IP Pakete sicher. Das Authentication Header Protokoll wird in dem RFC 2402 beschrieben. Dieser löst den älteren RFC 1826 ab.

Für seine Funktion setzt das AH Protokoll einen Integrity-Check-Value ein. Hierbei handelt es sich um die 96 höchstwertigen Bits eines Hash-Message Authentication Codes. Üblicherweise werden als HMAC sowohl HMAC-MD5-96 und HMAC-SHA-96 unterstützt. Neuere IPsec Varianten setzen jedoch auch HMAC-SHA-256-96, HMAC-SHA-384-96, HMAC-SHA-512-96 und HMAC-RIPEDM-160-96 ein. Als zusätzlichen Schutz versieht das AH Protokoll jedes Paket mit einer Sequenznummer, die vom Empfänger zum Schutz vor Replay-Angriffen genutzt werden kann.

Der Header des AH Protokolls ist in Abbildung 3.5 dargestellt.

Der Header enthält folgende Informationen:

- **Next Header** Dieses Feld (8 Bit) gibt das Protokoll der im Paket übertragenen Informationen an. Handelt es sich um ein Paket im Transport Modus, so befindet sich hier die Protokollnummer des entsprechenden höheren Protokolls, zum Beispiel 6 (TCP) und 17 (UDP). Wird gleichzeitig auch das ESP Protokoll eingesetzt, so befindet sich hier die Nummer 50 (ESP). Im Tunnel Modus ist hier bei Verwendung von IPv4 die Nummer 4 zu finden.
- **Payload Length** Hier wird die Länge des Headers in 8 Bit angegeben. Der Name Payload Length ist daher irreführend. Außerdem handelt es sich um einen IPv6 Header. Daher wird die Headerlänge in Vielfachen von 32 Bit angegeben, nachdem zuvor 64 Bit abgezogen wurden. Bei einem Header aus  $3 \cdot 32$  Bit plus 96 Bit ICV ergibt sich 192 Bit Gesamtlänge. Abzüglich 64 Bit bleiben 128 Bit geteilt durch 32 Bit ergibt einen Wert von 4 für die Payload Length (siehe RFC 2402).
- **Reserved** Dieses 16 Bit Feld ist für zukünftige Zwecke reserviert und muss mit Nullen aufgefüllt werden.
- **Security Parameter Index (SPI)** Diese 32 Bit Zahl identifiziert in Kombination mit der Ziel-IP Adresse und dem IPsec Protokoll (AH oder ESP) eindeutig die Security Association. Die Zahl 0 ist für interne Verwendung reserviert. Die Zahlen 1 bis 255 (0x1-0xff) sind für die Verwendung durch die Internet Assigned Numbers Authority (IANA) reserviert.

- **Sequence Number** Die 32 Bit lange Sequenznummer ist eine monoton steigende Nummer, die vom Absender jedem Paket zugewiesen wird. Der Empfänger kann diese Nummer nutzen, um sich vor Replay-Angriffen zu schützen.
- **Integrity Check Value (ICV)** In diesem 96 Bit langen Feld werden die Authentifizierungsdaten gespeichert. Diese Daten können mit den verschiedenen HMAC Verfahren erzeugt werden. In allen Fällen werden nur die 96 höchstwertigen Bits des ermittelten Hashes hier abgespeichert.

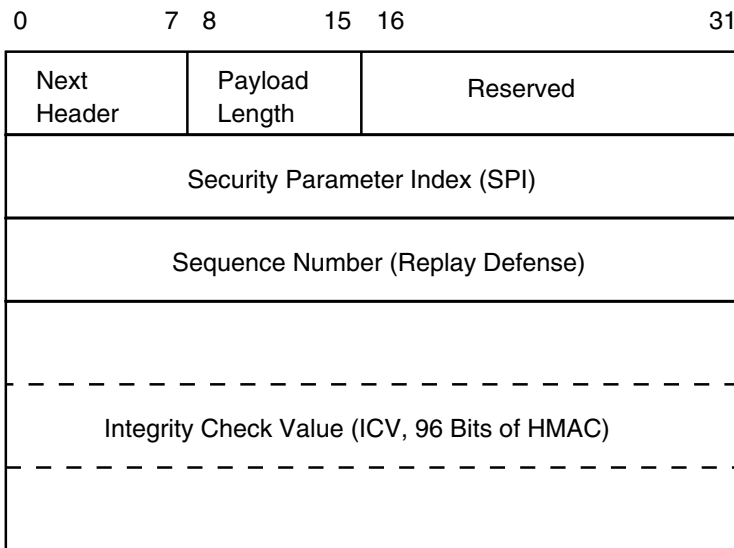


Abbildung 3.5 Das Authentication Header Protokoll hat einen 24 Byte langen Header

Das Authentication Header Protokoll sichert das gesamte IP-Paket. Das bedeutet, dass bei der Berechnung des ICVs nicht nur der AH Header und die gekapselten Informationen herangezogen werden, sondern auch der äußere IP-Header berücksichtigt wird. Dieser äußere IP-Header fließt in die Berechnung des ICV mit ein. Nicht berücksichtigt werden lediglich die folgenden Felder des IPv4-Headers: Type of Service (TOS), Flags, Fragment Offset, Time to Live (TTL) und Header Checksum. Diese Felder werden für die Berechnung des ICV auf Null gesetzt. Im Falle von IPv6 werden die folgenden Felder nicht berücksichtigt: Class, Flow Label, Hop Limit.

## ACHTUNG

Die Einbeziehung des äußeren IP-Headers und damit auch der IP Adressen in die Berechnung des ICV schließt die gleichzeitige Verwendung von Network Address Translation (NAT) aus. Eine Modifikation der IP Adressen nach Berechnung des ICV würde zu dessen Ungültigkeit führen.

### 3.2.6 Encapsulated Security Payload – ESP

Das Encapsulated Security Payload Protokoll (IP Protokoll 50) stellt die Authentifizierung, die Integrität und die Vertraulichkeit der IP Pakete sicher. Das ESP Protokoll wird in dem RFC 2406 beschrieben. Dieser löst den älteren RFC 1827 ab.

Für seine Funktion setzt das ESP Protokoll einen Integrity Check Value ein. Hierbei handelt es sich um 96 höchstwertige Bits eines Hash-Message Authentication Codes. Üblicherweise werden als HMAC sowohl HMAC-MD5-96 und HMAC-SHA-96 unterstützt. Neuere IPsec Varianten setzen jedoch auch HMAC-SHA-256-96, HMAC-SHA-384-96, HMAC-SHA-512-96 und HMAC-RIPEDM-160-96 ein. Als zusätzlichen Schutz versieht das AH Protokoll jedes Paket mit einer Sequenznummer, die vom Empfänger zum Schutz vor Replay-Angriffen genutzt werden kann. Um die Vertraulichkeit zu gewährleisten werden die zu schützenden Informationen zusätzlich verschlüsselt.

Der Header des ESP Protokolls ist in Abbildung 3.6 dargestellt.

Der Header enthält folgende Informationen:

- **Security Parameter Index (SPI)** Diese 32 Bit Zahl identifiziert in Kombination mit der Ziel-IP Adresse und dem IPsec Protokoll (AH oder ESP) eindeutig die Security Association. Die Zahl 0 ist für die interne Verwendung reserviert. Die Zahlen 1 bis 255 (0x1-0xff) sind für die Verwendung durch die Internet Assigned Numbers Authority (IANA) reserviert.
- **Sequence Number** Die 32 Bit lange Sequenznummer ist eine monoton steigende Nummer, die vom Absender jedem Paket zugewiesen wird. Der Empfänger kann diese Nummer nutzen, um sich vor Replay-Angriffen zu schützen.

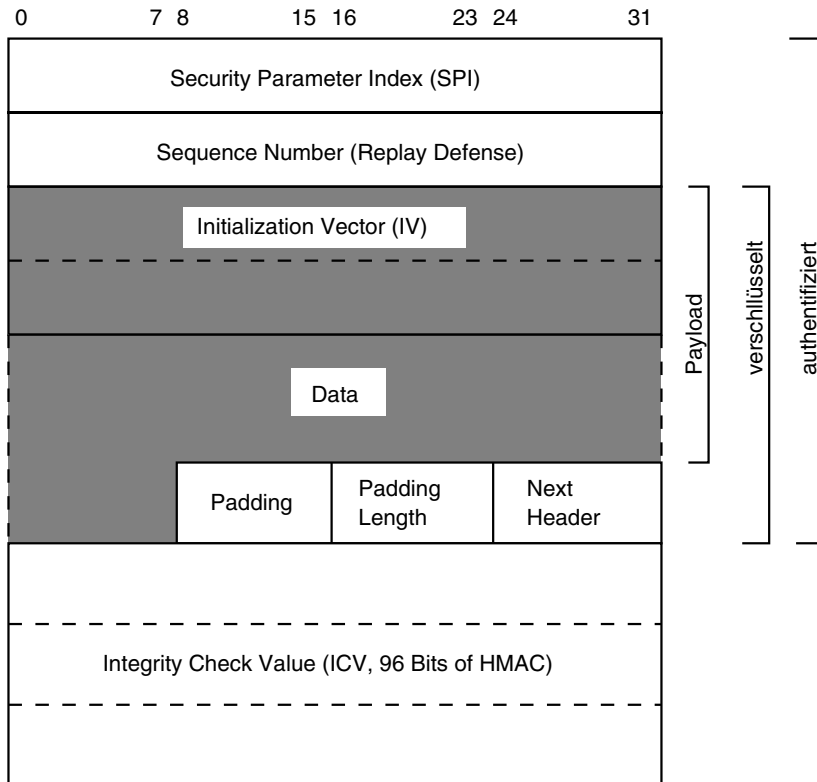


Abbildung 3.6 Das Encapsulated Security Payload Protokoll hat einen geteilten Header, der die zu schützenden Informationen umschließt

- **Payload** Hier werden die eigentlichen meist verschlüsselten Nutzinformationen abgespeichert. Wenn der Verschlüsselungsalgorithmus einen Initialization Vector benötigt, wird dieser zu Beginn abgespeichert.
  - **Initialization Vector (IV)** Sämtliche eingesetzten symmetrischen Verschlüsselungsverfahren sind monoalphabetisch und werden daher im Cipher-Block-Chaining Modus eingesetzt. Hierfür ist ein Initialisierungsvektor notwendig. Dieser wird zu Beginn abgespeichert. Die Länge richtet sich nach dem eingesetzten Verschlüsselungsverfahren. 3DES benötigt zum Beispiel einen 64 Bit langen IV.
  - **Data** Direkt im Anschluss an den IV werden die verschlüsselten Daten abgespeichert.
- **Padding** Da grundsätzlich bei IPsec Block-Ciphern eingesetzt werden, ist häufig ein Padding erforderlich. Hiermit werden die Daten bis zur nächsten Blockgrenze aufgefüllt. Das Padding kann 0 bis 255 Bytes lang sein.

- **Padding Length** Um nach der Entschlüsselung das Pad entfernen zu können wird hier dessen Länge abgespeichert.
- **Next Header** Dieses Feld (8 Bit) gibt das Protokoll der im Paket übertragenen Informationen an. Handelt es sich um ein Paket im Transport Modus, so befindet sich hier die Protokollnummer des entsprechenden höheren Protokolls, zum Beispiel 6 (TCP) und 17 (UDP). Im Tunnel Modus ist hier bei Verwendung von IPv4 die Nummer 4 zu finden.
- **Integrity Check Value (ICV)** In diesem 96 Bit langen Feld werden die Authentifizierungsdaten gespeichert. Diese Daten können mit den verschiedenen HMAC Verfahren erzeugt werden. In allen Fällen werden nur die 96 höchstwertigen Bits des ermittelten Hashes hier abgespeichert.

Bei der Verarbeitung der Daten durch das ESP Protokoll, werden zunächst die Informationen verschlüsselt und anschließend erst die Authentifizierungsinformationen (ICV) berechnet. Dadurch ist sichergestellt, dass der Empfänger die Daten nicht aufwändig entschlüsseln muss, wenn die Überprüfung der Authentifizierung fehlschlägt.

Bei der Berechnung des ICV wird im Gegensatz zum AH-Protokoll nicht der äußere IP-Header mit einbezogen. Der ICV bezieht sich lediglich auf den ESP-Header und die zu sichernden Daten. Daher ist es grundsätzlich möglich die IP Adressen im äußeren IP-Header auszutauschen (Network Address Translation, NAT) ohne dass der ICV ungültig wird. In vielen Fällen ist ein NAT dennoch nur eingeschränkt möglich, da die meisten NAT-Geräte eine interne Zuordnung der »genetteten« Verbindungen über die in der Verbindung verwendeten Ports durchführen. Sowohl das AH als auch das ESP Protokoll benutzen jedoch keine Ports.

### 3.2.7 Security Association

Die Security Association (SA, dt. Sicherheitsassoziation, RFC 2401) definiert die von den IPsec Protokollen zu verwendenden Parameter: Ziel-IP Adresse, IPsec-Protokoll, Verschlüsselungsalgorithmus, Authentifizierungsalgorithmus und Schlüssel. Zusätzlich wird eine Lebensdauer, der Modus (Transport Modus/Tunnel Modus) und weitere Eigenschaften (zum Beispiel Anti Replay Service (ARS)) abgespeichert.

Die Security Associations sind immer unidirektional. Das bedeutet, dass in der Praxis immer für eine bidirektionale Kommunikation zwei unidirektionale IPsec SA existieren müssen. Eine bestimmt den ausgehenden Verkehr und eine weitere den ankommenden Verkehr.

Ein Zugriff auf die Security Associations ist eindeutig möglich mit dem Triplet: SPI, Ziel-IP Adresse und IPsec-Protokoll. Daher können alle Security Associations in einer Security Association Datenbank (SAD) gespeichert werden.

Eine Security Association enthält mindestens die folgenden Parameter:

- Ziel-IP Adresse
- IPsec Protokoll
- Security Parameter Index (SPI)
- aktuelle Sequenznummer
- Definition für den Fall eines Sequenznummerüberlaufs.
- Anti-Replay-Fenster
- AH/ESP Algorithmus mit Schlüsseln
- Lebensdauer der SA
- IPsec Modus (Tunnel Modus/Transport Modus/Beliebig<sup>1</sup>)
- Path MTU

### 3.2.8 Security Policy

Die Security Association alleine führt noch nicht zu einer Verschlüsselung und/oder Authentifizierung des Netzwerkverkehrs. Sie spezifiziert lediglich, wie der Verkehr geschützt werden soll, aber nicht was und wann. Dies ist die Aufgabe der Security Policy (SP, RFC 2401).

Die Security Policies werden in der Security Policy Datenbank (SPD) gespeichert. Für jedes ein- wie ausgehende Paket wird die SPD konsultiert, ob dieses Paket in irgendeiner Form modifiziert werden muss. Die SPD liefert für jedes Paket eine von drei möglichen Antworten: DISCARD (verwerfen), PASS (unverändert durchlassen) und APPLY (IPsec SAs anwenden).

#### TIPP

Die DISCARD Funktion führt dazu, dass unter Windows 2000 mit Hilfe der Security Policies auch Firewallfunktionen realisiert werden. Viele Microsoft Windows Administratoren bringen daher IPsec nur mit Firewallfunktionen in Verbindung.

1. Achtung: Der Linux Kernel 2.6 unterstützt bisher keine SAs, die sowohl im Tunnel- als auch im Transport Modus verwendet werden können.

Verlangt die Security Policy die Anwendung einer IPsec SA, so muss sie zusätzlich diese IPsec SA spezifizieren.

Um die Gestaltung der Security Policies möglichst flexibel zu ermöglichen existieren die IPsec Selektoren. Sie erlauben es in der IPsec Security Policy spezifisch den Dienst zu definieren, der mit IPsec geschützt werden soll. Hierzu können die IP Adressen und das höhere Transportprotokoll (zum Beispiel TCP oder UDP) angegeben werden. Wenn das Transportprotokoll Ports unterstützt, können sie zusätzlich angegeben werden.

Hiermit ist es möglich nur den Verkehr eines NTP-Zeitserverns zu schützen. Das Network Time Protokoll verwendet den UDP Port 123.

**ACHTUNG**

Achtung! FreeS/WAN unterstützt nur mit den aktuellen Versionen des X.509 Patches Port- und Protokollselektoren. `racoon` und `isakmpd` unterstützen mit dem Linux Kernel 2.6 immer diese Selektoren.

### 3.2.9 Internet Key Exchange – IKE

Das Internet Key Exchange (IKE) Protokoll (RFC 2409) basiert auf dem Internet Security Association Key Management Protokoll (ISAKMP, RFC 2408), dem Oakley Key Determination Protokoll (RFC 2412), der IPsec Domain of Interpretation (DOI, RFC 2407) und dem SKEME Protokoll.

Hier soll aus Platzgründen lediglich das IKE Protokoll betrachtet werden. Weitere Ausführungen sind in den RFCs und in den Büchern [Lipp2001] und [Doroswamy1999] zu finden.

Das IKE Protokoll ermöglicht einen sicheren authentifizierten Schlüsselaustausch und die Aushandlung von IPsec Security Associations. Hiermit ermöglicht es eine automatische Erzeugung der IPsec SAs und einen automatischen Aufbau der VPN Verbindungen. Zusätzlich kann es die Sicherheit der VPN Verbindung dauerhaft gewährleisten, da es in der Lage ist bei Ablauf einer SA diese mit neuen Schlüsseln neu zu erzeugen.

Im Gegensatz zum AH und dem ESP Protokoll handelt es sich bei IKE nicht um ein eigenständiges IP Protokoll. Es setzt vielmehr auf dem UDP Protokoll auf und verwendet üblicherweise den Port 500.

Das IKE Protokoll arbeitet hierzu in zwei Phasen. In Phase 1 handeln die beiden Kommunikationspartner eine ISAKMP Security Association (SA) aus. Sie stellt einen sicheren authentifizierten Kanal dar, über den alle weiteren

Verhandlungen erfolgen. Hierzu definiert das IKE Protokoll zwei verschiedene Modi: Main Modus und Aggressive Modus. Diese Modi unterscheiden sich in ihrem Aufwand und in ihrer Sicherheit (s.u.).

In Phase 2 wird vom IKE Protokoll der Quick Modus verwendet. Dieser sehr schnelle Modus greift auf die ISAKMP SA der Phase 1 zurück, nutzt deren sicheren Kanal und braucht daher nicht erneut die Authentifizierung durchzuführen. Mit dem Quick Modus erzeugt das IKE Protokoll die IPsec SAs.

Der New Group Modus stellt eine Modus dar, der zwischen Phase 1 und Phase 2 verwendet werden kann, um eine neue Diffie Hellmann (Oakley) Gruppe zu vereinbaren.

### **Main Modus**

In Phase 1 des IKE Protokolls wird eine ISAKMP SA ausgehandelt. Hierzu stehen zwei verschiedene Modi zur Verfügung: Main Modus und Aggressive Modus. Hier soll zunächst der Main Modus mit seinen Austauschvorgängen beschrieben werden. Sie unterscheiden sich jedoch stark je nach eingesetztem Authentifizierungsverfahren. Daher werden die Authentifizierungsverfahren einzeln betrachtet.

#### *Authentifizierung mit RSA-Signaturen*

Die Authentifizierung mit einer digitalen Signatur erfordert sechs Nachrichten zwischen dem Initiator und dem Empfänger (Responder). Die Abfolge der Nachrichten ist in Abbildung 3.7 dargestellt.

Der Initiator sendet zunächst in seinem ersten ISAKMP Paket ein oder mehrere Vorschläge (Proposal) für die ISAKMP SA. Diese Vorschläge enthalten die angebotenen und unterstützten Authentifizierungsalgorithmen, Verschlüsselungsalgorithmen und Oakley Gruppen. Der Responder wählt einen dieser Vorschläge aus und bestätigt diesen in der zweiten Nachricht. Lehnt der Responder sämtliche Vorschläge ab, so endet die Kommunikation bereits hier und die Verbindung kommt nicht zustande.

Anschließend sendet der Initiator in der dritten Nachricht das öffentliche Ergebnis seiner Diffie Hellmann Berechnung als  $K_{Ei}$  und einen zufälligen Nonce  $N_i$ , der vom Responder für die Berechnung der Signatur verwendet wird. Wenn der Initiator nicht den öffentlichen Schlüssel des Responders in Form eines Zertifikates besitzt, kann er es in dieser Nachricht anfordern.

Der Responder antwortet in der vierten Nachricht mit den analogen Informationen.

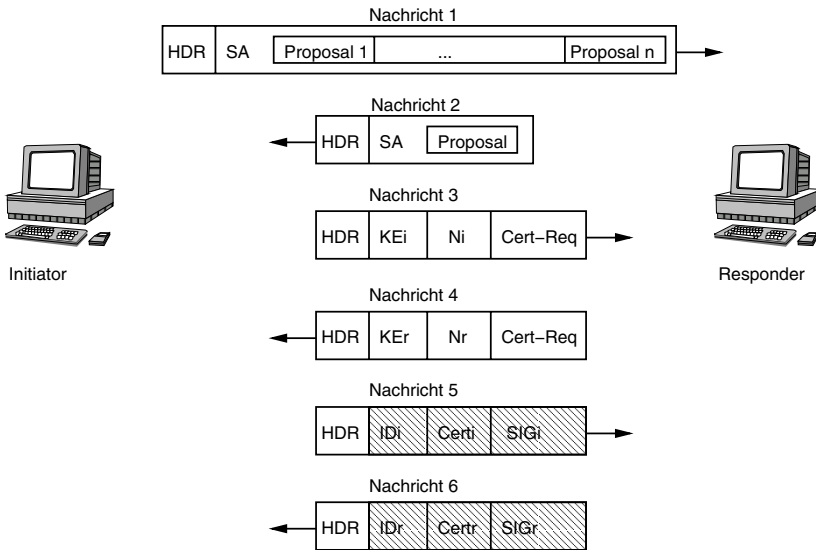


Abbildung 3.7 Die Authentifizierung mit digitalen Signaturen erfordert sechs Nachrichten im IKE Main Modus

Der Initiator kann nun mit dem Wert  $KEr$  die Diffie Hellmann Berechnung zu Ende führen und erhält einen symmetrischen Schlüssel. Er berechnet nun die digitale Signatur und sendet sie zusammen mit seiner Identität und bei Bedarf seinem Zertifikat an den Responder. Dabei werden diese Daten bereits mit dem symmetrischen Schlüssel verschlüsselt (schraffiert). Ein Dritter ist nicht in der Lage die Identität oder das Zertifikat unverschlüsselt zu lesen. Der Main Mode bietet hier einen Identitätsschutz.

Der Responder antwortet mit den analogen Informationen in der sechsten Nachricht.

Sowohl Initiator als auch Responder können die Inhalte der Nachrichten fünf und sechs entschlüsseln und die Signaturen mit den öffentlichen Schlüsseln überprüfen. Wurden diese in Form von Zertifikaten übermittelt, so müssen zuvor die Zertifikate auf ihre Korrektheit überprüft werden.

Dieses Verfahren wird meist eingesetzt, wenn X.509 Zertifikate genutzt werden.

### *Authentifizierung mit Public Key Verschlüsselung*

Bei der Authentifizierung mit Public Key Verschlüsselung werden RSA Schlüssel eingesetzt. Hierzu ist es erforderlich, dass die öffentlichen RSA Schlüssel des jeweiligen Partners im Vorfeld ausgetauscht wurden. Ein Austausch während Phase 1 ist nicht vorgesehen.

Nachricht 1 und 2 unterscheiden sich nicht von der Authentifizierung mit digitalen Signaturen.

In der Nachricht drei überträgt der Initiator wie gehabt auch das öffentliche Ergebnis seiner Diffie Hellmann Berechnung. Zusätzlich überträgt er seinen Nonce und seine Identität (Abbildung 3.8). Diese verschlüsselt er einzeln mit dem öffentlichen Schlüssel des Empfängers (schraffiert). Verfügt er über mehrere öffentliche Schlüssel des Empfängers, so wählt er einen aus und überträgt zusätzlich den Hash-Wert dieses Schlüssels ( $H(\text{Cert})$ ).

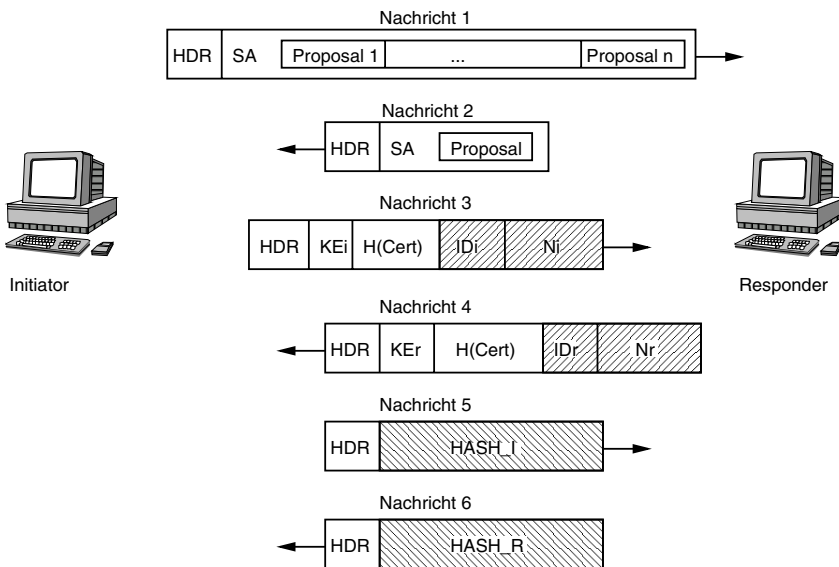


Abbildung 3.8 Die Authentifizierung mit Public Key Verschlüsselung erfordert vier Public Key Operationen

Der Responder antwortet analog in Nachricht vier. Insgesamt werden also vier Informationen  $ID_i$ ,  $N_i$ ,  $ID_r$  und  $N_r$  mit RSA verschlüsselt.

Nun können Initiator und Responder mit ihren privaten RSA Schlüsseln die Identität des Partners und dessen Nonce entschlüsseln. Zusätzlich können die Diffie Hellmann Berechnungen mit den Werten  $KE$  zu Ende geführt und der symmetrische Schlüssel bestimmt werden.

Initiator und Responder berechnen nun aus dem Nonce und weiteren Informationen einen Hash, der anschließend mit dem symmetrischen Schlüssel übertragen wird. Kann die jeweilige Gegenstelle den Hash authentifizieren, so besaß der Erzeuger des Hash den Nonce. Um den Nonce zu besitzen musste dieser jedoch zuvor mit dem privaten Schlüssel entschlüsselt werden.

Somit ist die Gegenseite authentifiziert. Dieses Verfahren wird nur selten eingesetzt.

### *Authentifizierung mit revidierter Public Key Verschlüsselung*

Die Authentifizierung mit Public Key Verschlüsselung bietet Vorteile, da ein Angreifer sowohl die symmetrische als auch die asymmetrische RSA Verschlüsselung knacken muss. Sie besitzt aber auch den Nachteil, dass vier Public Key Operationen pro System durchzuführen sind: zwei Verschlüsselungen und zwei Entschlüsselungen. Das revidierte Verfahren reduziert dies auf zwei Public Key Operationen.

Das revidierte Verfahren (Abbildung 3.8) ähnelt dem Public Key Verschlüsselungsverfahren. Die erste und zweite Nachricht sind wieder identisch. In Nachricht drei überträgt der Initiator den Nonce verschlüsselt mit dem Public Key des Responders. Erneut muss er den Hash des verwendeten Public Keys übertragen, wenn mehrere Public Keys zur Verfügung stehen.

Das Ergebnis der Diffie Hellmann Berechnung, seine eigene Identität und ein möglicher eigener Public Key werden jedoch nicht mit dem Public Key des Empfängers verschlüsselt, sondern mit einem symmetrischen Schlüssel. Da das Diffie Hellmann Verfahren aber noch nicht abgeschlossen wurde, ermittelt der Initiator diesen Schlüssel aus seinem eigenen Nonce. Der Responder kann den Nonce mit seinem privaten Schlüssel dekodieren und den symmetrischen Schlüssel nach dem identischen Verfahren ableiten. Mit diesem kann er dann die Daten  $ID_i$  und  $KE_i$  entschlüsseln.

Der Responder antwortet wieder mit analogen Daten.

Die beiden Nachrichten fünf und sechs enthalten wieder Hashes, die neben anderen Informationen auch auf den ausgetauschten Nonces beruhen. Sie dienen der Authentifizierung.

### *Authentifizierung mit Preshared Keys (PSK)*

Die Authentifizierung mit einem Preshared Key ist ebenso in der Phase 1 des IKE Main Modus möglich. Hierzu werden die beiden ersten Nachrichten wie bei den anderen Methoden ausgetauscht (Abbildung 3.9).

Anschließend tauschen Initiator und Responder den Nonce und das öffentliche Ergebnis der Diffie Hellmann Berechnung aus (Nachrichten drei und vier). Dabei wird der PSK bereits für die Erzeugung der Ausgangswerte für die Schlüsselberechnung verwendet.

Nach dem Austausch der Werte  $KE$  können Initiator und Responder die Nachrichten fünf und sechs übermitteln. Hier werden unter anderem die Identitäten verschlüsselt ausgetauscht.

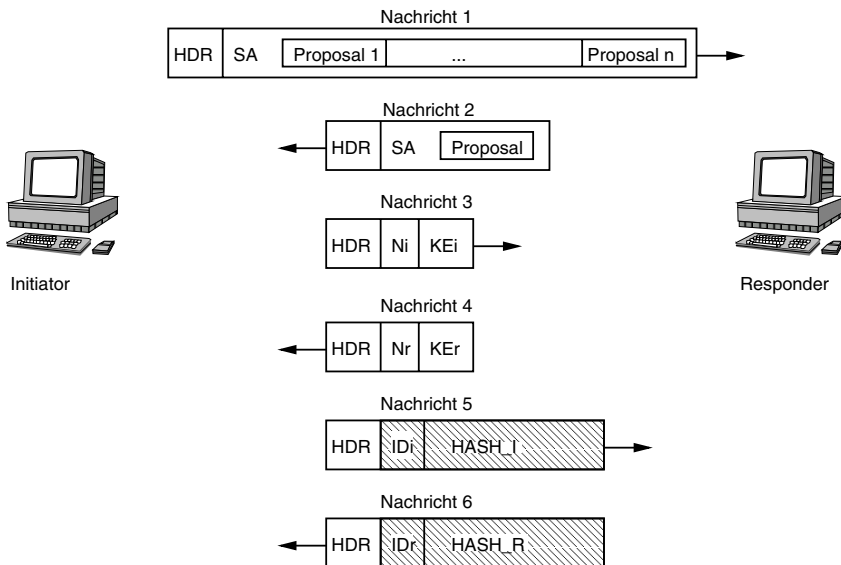


Abbildung 3.9 Bei der Authentifizierung mit Preshared Keys wird die Identität erst in den letzten Nachrichten verschlüsselt übertragen

Dies ist der wesentliche Grund, warum bei der Verwendung von PSKs im Main Mode bei unbekanntem IP-Adressen alle Kommunikationspartner denselben PSK verwenden müssen. Werden unterschiedliche PSKs benutzt, so muss der Kommunikationspartner bereits vor dem Senden der Nachrichten drei und vier den richtigen PSK wählen. Da die Identität noch nicht übertragen wurde (Identitätsschutz des Main Modus) kann die Auswahl der PSKs nur auf der IP-Adresse beruhen. Ist auch die unbekannt, kann nur ein allgemeiner PSK verwendet werden.

### Aggressive Modus

Der Aggressive Modus stellt eine schnellere Variante zur Verhandlung der Phase 1 dar. Hierbei werden sämtliche Informationen in nur drei Nachrichten ausgetauscht. Dadurch sind im Gegensatz zum Main Modus Denial-of-Service-Angriffe möglich, da bereits das erste Paket eine Verschlüsselung verlangt. Ein Angreifer kann so eine Vielzahl von Paketen erzeugen und durch unzählige sinnlose Verschlüsselungsvorgänge einen DoS erzeugen.

Der zweite Unterschied beim Aggressive Modus betrifft die Authentifizierung mit Preshared Keys und digitalen Signaturen. Werden PSKs oder digitale Signaturen im Aggressive Modus eingesetzt, so wird die Identität im Klartext übertragen. Der Aggressive Modus verlangt keinen Identitätsschutz wie der Main Modus. Daher ist es möglich im Aggressive Modus unter-

schiedliche PSKs auch bei dynamischen unbekanntenen IP Adressen einzusetzen. Bei der Authentifizierung mit Public Key Verschlüsselung werden weiterhin die Identitäten geschützt.

Der fehlende Identitätsschutz des Aggressive Modus stellt eine Sicherheitslücke dar. Es existieren einige Werkzeuge (IKEcrack, <http://sourceforge.net/projects/ikecrack>) und Artikel ([www.ernw.de/download/pskattack.pdf](http://www.ernw.de/download/pskattack.pdf)), die diese Sicherheitslücke beschreiben und ausnutzen können.

Eine Verwendung des Aggressive Modus sollte aus diesen Gründen nur in Betracht gezogen werden, wenn der Main Modus von einem Kommunikationspartner nicht unterstützt wird.

### Quick Modus

Der Quick Modus wird in der Phase 2 vom IKE Protokoll verwendet. Er setzt immer eine erfolgreich durchlaufene Phase 1 voraus. Hiermit können die IPsec Security Associations auf der Basis der ISAKMP SA der Phase 1 erzeugt werden. Dabei können unter dem Schutz einer ISAKMP SA mehrere IPsec SAs sogar innerhalb eines Quick Mode erzeugt werden. Hier wird nun der Vorteil der Aufteilung des IKE Protokolls in Phase 1 und Phase 2 deutlich. Werden mehrere IPsec SAs benötigt, so müssen die aufwändigen Authentifizierungsnachrichten nur einmalig beim Aufbau der ISAKMP SA ausgetauscht werden. Die ISAKMP SA stellt damit auch schon Dienste zur Überprüfung der Integrität, Authentifizierung und zur Sicherstellung der Vertraulichkeit zur Verfügung.

Sämtliche Pakete der Phase 2 werden verschlüsselt ausgetauscht. Für den Aufbau einer IPsec SA werden drei Nachrichten benötigt (Abbildung 3.10).

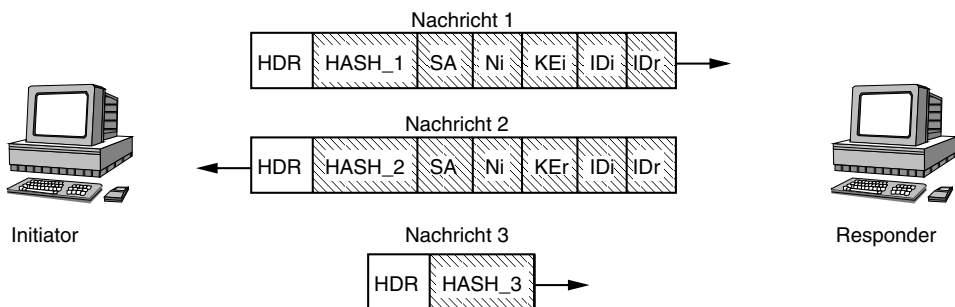


Abbildung 3.10 Der Quick Mode überträgt alle Informationen verschlüsselt

Hierbei müssen der SA Vorschlag, der Hash-Wert und der Nonce-Wert übertragen werden. Optional sind die Identitäten und der Diffie Hellman Wert. Der Diffie Hellman Wert wird benötigt, wenn Perfect Forward Secrecy gewünscht wird. Dann werden sämtliche benötigten Schlüssel neu berechnet und nicht von den Masterschlüsseln der ISAKMP SA abgeleitet. Damit wird sichergestellt, dass ein Angreifer durch die Berechnung eines Schlüssels nicht weitere Schlüssel erschließen kann.

### 3.2.10 UDP Encapsulation

Einer der wichtigsten Beweggründe für die Entwicklung des IPv6 Protokolls war die prognostizierte Knappheit der IPv4 Adressen zu Beginn der 90er Jahre. Das IPv6 Protokoll bietet mit einem 128 Bit großen Adressraum ausreichend IP Adressen »für jeden Toaster und jede Ampel auf der Erde«.

Dies führte jedoch auch dazu, dass bei der Entwicklung der IPsec Protokolle nicht berücksichtigt wurde, dass Network Address Translation (NAT) erforderlich sein könnte. Network Address Translation wurde entwickelt, um der Knappheit der IPv4 Adressen entgegenzutreten. So benötigt ein großes Unternehmen mit mehreren Tausenden Rechnern nur wenige offizielle IPv4 Adressen um jedem Rechner einen Zugriff aufs Internet zu ermöglichen und erreichbar zu sein. Intern werden private IPv4 Adressen verwendet, die beim Zugriff auf das Internet meist von einer Firewall in offizielle IPv4 Adressen umgesetzt werden.

Das Authentication Header (AH) Protokoll verhindert diese Umsetzung wirkungsvoll, da es auch die IP Adressen des äußeren IP-Headers authentifiziert. Werden sie verändert, so wird das Paket ungültig. Das Encapsulated Security Payload weist diese Einschränkung nicht auf. Dennoch führt sein Einsatz in NAT Umgebungen häufig zu Problemen.

Um dies zu verstehen soll kurz vorgestellt werden, wie in den meisten Fällen das NAT durchgeführt wird. Hierbei wird nur auf das Source NAT, das Austauschen der Absender IP Adresse eingegangen. Jedoch arbeitet das Destination NAT, das die Ziel IP Adresse modifiziert, analog.

Die Abbildung 3.11 zeigt die Funktion des NAT Gateways.

Ein NAT Gerät erzeugt eine Tabelle, in der sämtliche Verbindungen gepflegt werden. Um einen eindeutigen Index auf diese Tabelle zu erhalten, werden die Verbindungen nach der Client IP Adresse und dem Client Port (bei TCP und UDP) sortiert. Um ankommende Antwort Pakete eines Servers eindeutig zuzuordnen zu können, wird jedem Paar aus Client IP Adresse und Port ein Paar aus IP Adresse des NAT Gateways und Port zugewiesen. Dies ist jedoch

nur möglich wenn das Protokoll Ports unterstützt. Bei einigen Protokollen (zum Beispiel ICMP) können auch andere Parameter des Protokolls genutzt werden.

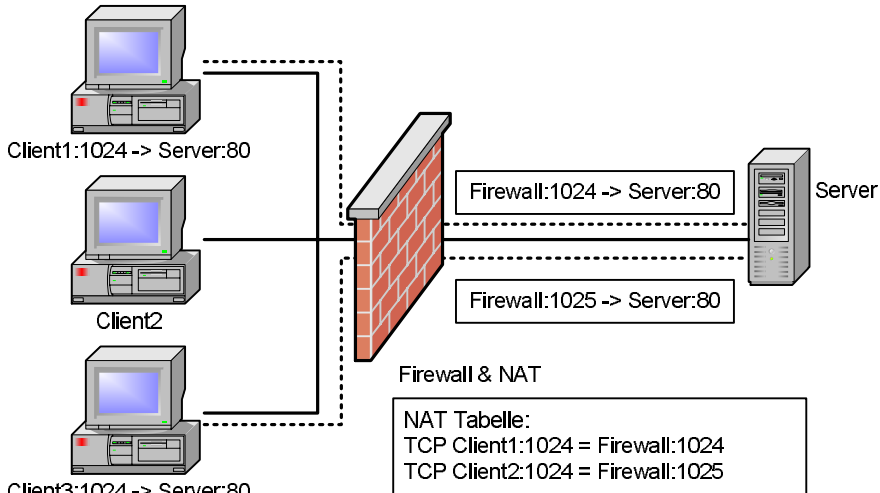


Abbildung 3.11 Ein NAT Gateway ordnet die Pakete den genatteten Verbindungen über die Client IP Adresse und den Client-Port zu

Das Protokoll ESP unterstützt nun jedoch keine Ports. Die einzige Information, die dem Port nahe kommt und in Klartext lesbar ist, ist der SPI. Jedoch unterstützen die wenigsten NAT Gateways den SPI in ihren NAT Tabellen.

Aus diesem Grund gibt es Probleme, sobald mehr als ein Rechner über ein NAT Gateway eine IPsec Verbindung aufbauen möchte. Das NAT Gateway hat Schwierigkeiten, die Antwortpakete den Clients wieder zuzuordnen zu können, da keine Unterscheidung der Verbindungen auf der Basis des Ports vorgenommen werden kann.

Hier hilft die erneute Kapselung aller IPsec Pakete in UDP Paketen. UDP Pakete weisen einen Port auf, und können daher durch eine Standard NAT Tabelle unterschieden werden.

Zwei Internet Engineering Task Force (IETF) Drafts beschreiben das sogenannte NAT Traversal: »Negotiation of NAT Traversal in the IKE« und »UDP Encapsulation of IPsec Packets«.

Das erste Draft beschreibt eine Erweiterung des IKE Protokolls. Damit ist es möglich, automatisch die Unterstützung von NAT Traversal durch den Kommunikationspartner und das Vorhandensein eines NAT Gerätes zwischen den Kommunikationspartnern zu erkennen.

Diese Erkennung erfolgt in Phase 1 des IKE Protokolles. Die NAT Traversal Unterstützung wird durch spezielle Vendor ID Nachrichten erkannt.

Die Lokalisierung eines möglichen NAT Gerätes erfolgt durch spezielle NAT Discovery (NAT-D) Pakete in Phase 1 (Abbildung 3.12).

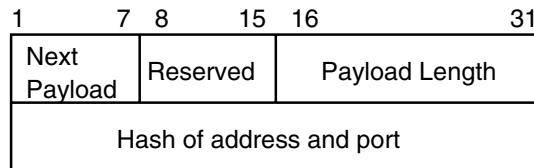


Abbildung 3.12 Das NAT Discovery Paket

Diese NAT D Pakete enthalten jeweils den Hash der Quell- und der Ziel IP Adresse und des Ports. Berechnet der Empfänger einen anderen Hash aufgrund der IP Adresse und des Ports im IP Header, so befindet sich ein NAT Gerät zwischen Absender und Empfänger. Diese Pakete sind beim Main Mode in den Nachrichten drei und vier und im Aggressive Mode in den Nachrichten zwei und drei enthalten. Verfügt ein Absender über mehrere IP Adressen über die das Paket den Rechner verlassen darf, so kann er mehrere dieser Pakete in seinen Nachrichten übertragen.

Da es einige NAT Geräte gibt, die versuchen mit IPsec intelligent umzugehen und dies beim NAT Traversal störend sein kann, sollte möglichst bald der IKE-Port 500/udp gewechselt werden. Diese intelligenten NAT Geräte erkennen den IPsec Verkehr meist nur am IKE-Port 500/udp. IKE Daemonen, die NAT Traversal unterstützen, verwenden daher, sobald sie NAT erkannt haben, den Quell- und den Ziel-Port 4500/udp.

Alle weiteren Rekey-Vorgänge der ISAKMP SA und der IPsec SAs werden mit diesem Port durchgeführt.

Hiermit ist Phase 1 abgeschlossen. Nun können in Phase 2 mit dem Quick Modus die IPsec SAs verhandelt werden. Hierbei stehen zwei neue Kapselmodi zur Verfügung:

- UDP-Encapsulated-Tunnel
- UDP-Encapsulated-Transport

Diese beiden Modi werden im Draft »UDP Encapsulation of IPsec Packets« beschrieben. Die wesentlichen Grundzüge werden im Folgenden dargestellt.

Die IPsec ESP Pakete werden in UDP Pakete eingepackt. Hierbei wird derselbe Port verwendet, wie er auch bereits für die IKE Verhandlungen verwendet wurde: 4500/udp. Der UDP ESP Header ist in Abbildung 3.13 dargestellt.

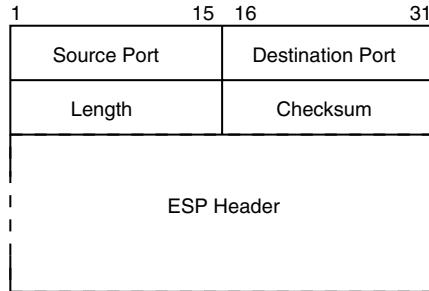


Abbildung 3.13 Das ESP Paket wird in einem UDP Header eingepackt

Damit die NAT Verbindung über NAT Geräte am Leben gehalten werden kann, auch wenn kein Verkehr zu transportieren ist, werden NAT Keepalive Pakete gesendet. Ansonsten verwerfen einige NAT Geräte bereits nach kurzer Zeit, nämlich 180 Sekunden, die Verbindung aus der NAT Tabelle. Der NAT Keepalive Header ist in Abbildung 3.14 zu sehen.

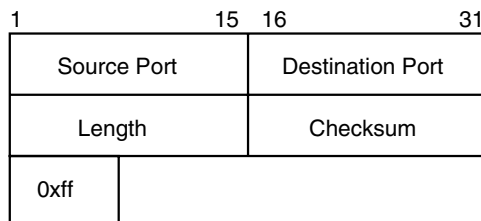


Abbildung 3.14 Der UDP NAT Keepalive Header hält die Verbindung aufrecht, wenn keine Daten transportiert werden müssen

Diese Keepalive Pakete werden normalerweise alle 20 Sekunden versandt, wenn kein anderes Paket über den Tunnel transportiert wurde.

### 3.2.11 DHCP-over-IPsec

In vielen Fällen, bei denen ein entfernter Rechner über eine VPN Verbindung mit einem LAN in Kontakt tritt, ist es von Vorteil, wenn anschließend der entfernte Rechner eine IP Adresse aus dem internen Netzwerk erhält und

sich scheinbar im LAN befindet. Dies kann erreicht werden, indem der Client eine virtuelle IP Adresse per DHCP erhält.

Es existieren eine Vielzahl von Drafts und ein RFC (RFC 3456), die den Einsatz von DHCP über IPsec beschreiben. Hier soll kurz die Technologie beschrieben werden, wie sie vom RFC 3456 als Standard vorgeschlagen wird. Dies weicht in einzelnen Punkten von den frühen Drafts und Implementierungen ab.

Die Verwendung von DHCP ist nur für IPv4 Adressen im IPsec Tunnel Modus beschrieben und sinnvoll. Hierbei kann mit DHCP die IP Adresse an den Client zugewiesen werden. Die Verwendung von DHCP bietet folgende Vorteile:

- Die Verwendung von DHCP erleichtert den Einsatz und die Integration in großen Netze, da diese meist sowieso bereits mit Hilfe des DHCP Protokolls verwaltet werden.
- Zusätzlich bietet das DHCP Protokoll die Möglichkeit den Adressen Pool anhand bestimmter Eigenschaften des Clients zu verwalten. So können bestimmten Clients bestimmte IP Adressen zugewiesen werden.
- Die Speicherung der DHCP Daten und der DHCP Datenbank auf dem DHCP Server erleichtert die Konfiguration einer Hochverfügbarkeitslösung, da diese Informationen nicht über die VPN-Gateways verteilt werden.
- Das DHCP Protokoll verfügt bereits über Verfahren zur Neukonfiguration der IP Adressen. Diese Funktion muss daher nicht im IKE Protokoll implementiert werden.
- Sämtliche DHCP Funktionen können ohne weitere Änderung des IKE Protokolls genutzt werden. Dadurch sind keine Einschränkungen der Sicherheit oder eine zusätzliche Komplexität zu erwarten.

Die Abbildung 3.15 zeigt eine typische Anwendung. Hierbei baut der externe Client einen IPsec Tunnel auf. Durch den IPsec Tunnel verbindet er sich mit einer virtuellen IP Adresse, die er zuvor vom DHCP Relay auf dem VPN Gateway erhalten hat. Anschließend befindet er sich scheinbar im internen Netzwerk. Hierzu benötigt der externe Rechner zwei IP Adressen und üblicherweise zwei Netzwerkkarten. Eine physikalische Netzwerkkarte mit echter IP Adresse wird verwendet, um den Tunnel zum VPN Gateway aufzubauen. Diese IP Adresse wird auch als IP Adresse im äußeren IP Header verwendet. Die zweite virtuelle Netzwerkkarte verwendet die virtuelle IP Adresse die über DHCP zugeteilt wurde. Diese IP Adresse wird im inneren, verschlüsselten IP Header des Tunnels verwendet.

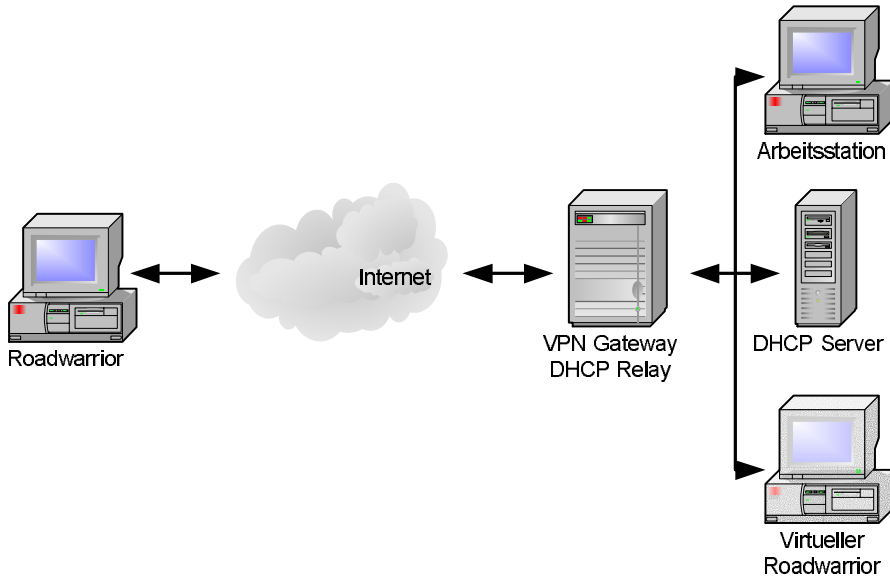


Abbildung 3.15 Der Client fordert zunächst per DHCP eine IP Adresse an, um später mit dieser Adresse über den Tunnel zu kommunizieren

Die DHCP-over-IPsec Kommunikation funktioniert folgendermaßen:

1. Der Client baut eine ISAKMP SA in der Phase 1 des IKE Protokolls auf.
2. Der Client baut eine DHCP SA mit dem IPsec Tunnel Mode VPN Gateway auf. Hier werden Protokoll- und Portselektoren verwendet, um sicherzustellen, dass lediglich DHCP Verkehr über diese SA ausgetauscht werden darf.
3. Die DHCP Nachrichten werden zwischen dem Client und dem DHCP Relay auf dem VPN Gateway ausgetauscht. Das DHCP Relay fordert die virtuelle IP Adresse für den Client vom DHCP Server an und gibt sie an den Client weiter.
4. Nun kann die Original DHCP SA gelöscht und eine neue SA mit der neuen IP Adresse aufgebaut werden (üblich) oder die vorhandene durch Erweiterung der Protokoll- und Portselektoren für den allgemeinen Verkehr geöffnet werden.

Obwohl sicher andere Varianten existieren, wird normalerweise das VPN Gateway nicht auch der DHCP Server sein. Daher ist auf dem VPN Gateway ein DHCP Relay erforderlich.

## TIPP

FreeS/WAN mit dem X.509 Patch kann als VPN Gateway diese Funktionen wahrnehmen. Der Linux Kernel 2.6 unterstützt mit `isakmpd` den IKE Config Modus.

### 3.3 L2TP

Das Layer-Two-Tunneling-Protokoll (L2TP) ist kein VPN Protokoll per se. Es bietet lediglich Tunnelfunktionen. Hierzu tunnelt es beliebige Pakete in einem UDP Tunnel. Es verwendet den Port 1701. Das L2TP ist nicht in der Lage die Integrität, Authentizität oder Vertraulichkeit der übertragenen Daten zu garantieren. Dennoch soll das Protokoll hier betrachtet werden, da verschiedene Hersteller, zum Beispiel Microsoft, es in Kombination mit dem IPsec Protokoll einsetzen um VPN-Funktionen anzubieten.

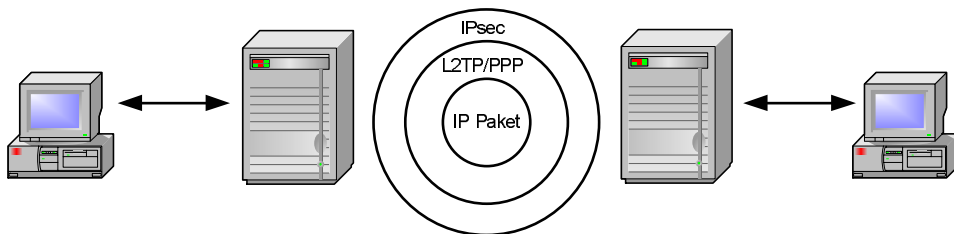


Abbildung 3.16 IPsec schützt den L2TP Tunnel

Hierbei wird zunächst mit dem IPsec Protokoll eine verschlüsselte Verbindung zwischen den Kommunikationspartnern aufgebaut. Diese Verbindung bietet nun garantiert die Authentizität, Integrität und Vertraulichkeit der übertragenen Informationen. Auf der Basis dieser Verbindung wird anschließend mit dem L2TP Protokoll ein weiterer L2TP Tunnel aufgebaut (Abbildung 3.16). Dieser Tunnel bietet alle Vorteile des L2TP Protokolls.

Im einzelnen sind das die folgenden Punkte:

- Der Aufbau des L2TP Tunnels erfordert eine erneute Benutzerauthentifizierung. Sie kann von der IPsec Authentifizierung verschieden sein.
- Der L2TP Tunnel kann einen anderen Endpunkt haben, als die IPsec Verbindung.
- Der L2TP Tunnel kann auch nicht IP-Pakete transportieren, zum Beispiel IPX. Es ist ein Tunnel auf Layer 2.

- Der L2TP Tunnel kann andere IP Adressen verwenden, als die IPsec Verbindung. Das L2TP Protokoll bietet sogar Funktionen, diese IP Adressen automatisch einem Client zuzuweisen (ähnlich DHCP over IPsec).

Damit ist das L2TP Protokoll in Kombination mit dem IPsec Protokoll ideal dazu geeignet, Rechnern mit dynamischer IP Adresse (Roadwarrior) von außen einen gesicherten Zugriff auf ein Unternehmensnetzwerk zu ermöglichen.

Die IPsec Protokolle wurden bereits ausführlich betrachtet. Für die Sicherung des L2TP Protokolls muss verpflichtend das ESP Transport Protokoll implementiert werden. Die Implementierung des ESP Tunnel Protokolls ist optional. Eine automatische Schlüsselverwaltung ist erforderlich. Das IKE Protokoll wird für diesen Zweck empfohlen. Diese Anforderungen werden von allen dem Autor bekannten Implementierungen eingehalten.

Jetzt soll das L2TP Protokoll kurz erläutert werden. Ausführlichere Informationen finden sich in den RFCs RFC 2661 (Layer Two Tunneling Protocol) und RFC 3193 (Securing L2TP using IPsec).

### 3.3.1 Einführung

Das L2TP Protokoll ist eine Erweiterung des Point-to-Point Protokolls (PPP). Das PPP Protokoll definiert die Schachtelung beliebiger Pakete über eine Layer 2 (L2) Point-to-Point Verbindung. Hierzu fordert der Benutzer zunächst eine L2 Verbindung an und benutzt anschließend PPP über diese Verbindung.

L2TP erlaubt dabei im Gegensatz zu PPP, dass die L2 Verbindung und die PPP Verbindung auf unterschiedlichen physikalischen Geräten enden.

Für die Beschreibung des L2TP Protokolls ist die Bedeutung einiger Begriffe erforderlich, von denen die wichtigsten im Folgenden beschrieben werden sollen:

- **CHAP** Challenge Handshake Authentication Protokoll (RFC 1994). Ein Authentifizierungsverfahren, bei dem das Kennwort nicht im Klartext übertragen wird.
- **Attribute Value Pair (AVP)** Ein Paar aus einem Attribut und einem Wert. Steuerungsmittelungen bestehen aus mehreren AVPs.
- **L2TP Access Concentrator (LAC)** Ein Endpunkt eines L2TP Tunnels.
- **L2TP Network Server (LNS)** Ein Endpunkt eines L2TP Tunnels und der Kommunikationspartner des LAC.



- **x** Reserviert für zukünftige Verwendung
- **S** Wenn dieses Feld gesetzt ist, enthalten die Pakete eine Sequenznummer
- **O** Wenn dieses Feld gesetzt ist, enthalten die Pakete ein Offsetfeld
- **P** Wenn dieses Feld gesetzt ist, sind die Pakete mit erhöhter Priorität zu behandeln.
- **Ver** L2TP Version. Dieses Feld muss den Wert 2 tragen für L2TP.
- **Length** Dieses optionale Feld enthält die Gesamtlänge in Bytes.
- **Tunnel ID** Dies kennzeichnet den einzelnen Tunnel. Diese Angabe ist nur lokal eindeutig. Die Endpunkte können unterschiedliche Tunnel IDs verwenden. Die Tunnel ID in einer Nachricht, ist immer die Tunnel ID des Empfängers.
- **Session ID** Dies kennzeichnet die einzelne Sitzung in einem Tunnel. Die in einer Nachricht gesetzte Session ID ist immer die ID des Empfängers.
- **Ns, Nr** Sequenznummern für gesendete Pakete (s) und empfangene Pakete (received, r).
- **Offset** Dieses Feld definiert, wo im L2TP Paket die eigentlichen Daten beginnen.

Diese Ausführungen sollen hier genügen, um das L2TP Protokoll einsetzen und analysieren zu können.