Diplomarbeit
1999

Virtual Private Network

Mit sicherem Tunnel durchs Internet

Olivier Gärtner
Berkant Uenal

Inhaltsverzeichnis

1 Zusammenfassung

Viele Firmen sind mit ihren Niederlassungen verbunden. Aus Kostengründen werden Verbindungen über das Internet den teuren Mietleitungen vorgezogen. Da das Internet ein öffentliches Netz ist und die Sicherheit bis anhin nicht relevant war, müssen heute Lösungen geboten werden, welche die Sicherheitsbedürfnisse der Firmen erfüllen. Mit dem Konzept eines ‘Virtual Private Network‘ (VPN) werden die Sicherheitslücken für Firmenanbindungen übers Internet weitgehend geschlossen.

Es war unsere Hauptaufgabe firmenbezogene Internet-Verbindungen zu realisieren. Dazu gehörte der Aufbau eines Versuchsnetzes in Linux-Umgebung, welche als Plattform für verschiedene VPN-Architekturen diente. In einem ersten Schritt wurden zwei Subnetze über zwei ‘Security Gateways‘ mit einem sicheren Tunnel miteinander verbunden. Diesbezüglich wurde das Sicherheits-Protokoll IPSec (Internet Protokoll Security) verwendet. Die wesentlichen Techniken des IPSecs (Tunneling, Chiffrierung und Authentisierung) wurden studiert. Im Rahmen unserer Diplomarbeit wurden die zwei Authentisierungsmethoden mit ‘Pre-Shared-Secrets‘ und digitale Signaturen (Zertifikate) beim Verbindungsaufbau näher analysiert und dokumentiert. Um X.509 Zertifikate selbst auszustellen, wurde eine Zertifizierungsstelle eingerichtet. Als nächstes wurde die Topologie des Versuchsnetzes für Remote Access ausgelegt, damit mobile Clients über einen Internet Service Provider eine VPN-Verbindung zum Security Gateway unseres Versuchsnetzes aufbauen können. Ein weiteres Schwergewicht legten wir auf die Paketanalyse der ausgetauschten Datenpakete mit Hilfe eines Netzwerkanalysators, der uns auch als Werkzeug zum Testen der Sicherheit diente.

Realisiert wurde ein lauffähiges VPN, das lokale Netzwerke sicher übers Internet verbindet. Im Hinblick auf eine spätere Realisation in verschiedenen Anwendungsgebieten wurden wichtige Erfahrungen und Wissen gewonnen.

Diese Arbeit war eine grosse Herausforderung für uns. Die Teamarbeit gestaltete sich höchst erfreulich, weil wir uns gut ergänzt haben. Wir sind mit dem Ergebnis der geleisteten Arbeit sehr zufrieden.

Winterthur, 1. Novermer 1999

Olivier Gärtner Berkant Ünal

2 Abstract

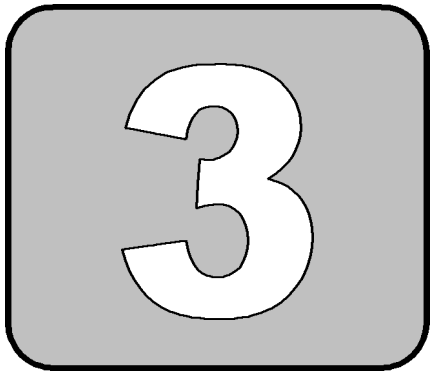
Companies are interested to have their establishments networked. Instead of using expensive leased lines, internet solutions are preferred. Since the internet is a public network not much attention was paid to security so far. Concepts of virtual private networks VPN meet the requirements to connect corporate networks securely over the internet.

The implementation of a virtual private network is considered as mean task of our thesis work. Hence an evaluation network in Linux environment has been set up, which served as platform for different VPN architectures. In a first step a site-to-site connection was built. Therefore two security gateways where configured to provide a secure tunnel. IPSec is an internet security protocol, which uses strong cryptography to provide authentication, encryption and tunneling services. All these features were in-depth studied.

In the scope of our thesis, the connection setup using the authentication methods pre-shared-secrets and digital signatures (certificates), were analyzed and documented. In order to issue and sign X.509-certificates a certification authority was installed.

Further more the network topology was adjusted for remote access, so that mobile clients can easily establish a secure connection to the corporate network. Our attention was also directed to analyze data packets of the exchanged traffic with the help of a network analyzer, which was also used to justify the provided security.

We succeeded in implementing a well working system, which allows you to build secure tunnels through untrusted networks. Regarding a later implementing in different areas of application important experiences and knowledge were gained.
The entire project was a great challenge for us. The teamwork became most pleasing, because we completed ourselves well. We are very content with the result of the work.



3 Aufgabenstellung

Zürcher Hochschule Winterthur

Schlussdiplomprüfung
Praktische Prüfung

Studiengang: **Elektrotechnik**

Jahr: 1999
Experten: Spühler / Uhlig

Studierende: **Olivier Gärtner / Berkant Uenal**
Klasse: ET5c / ET5d

Dozentin/Dozent: A. Steffen

Ausgabe der Aufgabe: **Do. 9. September 1999**

Zeitaufwand:

Abgabetermin: **Mo. 1. November 1999 12:50 h**

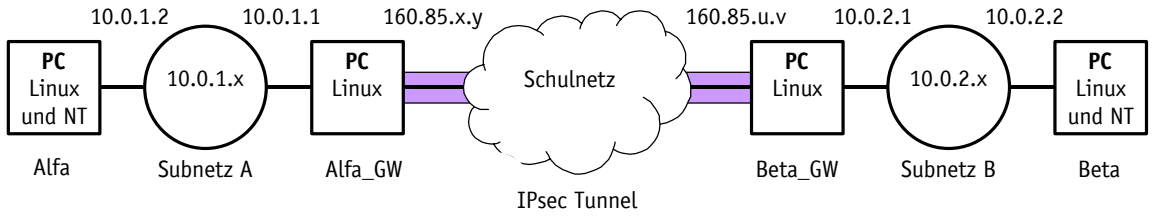
Thema: Virtual Private Network mit sicherem Tunnel durchs Internet

Es soll das im Rahmen des weltumspannenden Linux FreeS/WAN Projekts entwickelte sichere Internet Protokoll "IPsec" auf zwei Linux-Rechnern installiert und mit diesen Security Gateways eine Kopplung von zwei privaten Netzwerken über einen verschlüsselten Internet-Tunnel geschaffen werden.
Grosses Gewicht soll auf eine flexible und automatisierte Verwaltung der benötigten Authentisierungs- und Chiffrierungsschlüssel gelegt werden. Eine Erweiterung auf eine Public Key Authentisierung von Hosts und Users mittels X.509 Zertifikaten (OpenSSL) ist anzustreben.

Spezifikationen

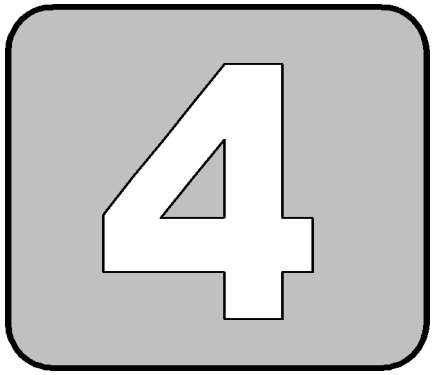
Allfällige Vorbereitungsarbeiten, Daten, Unterlagen, Literatur,
Hilfsmittel, Laboreinrichtungen, Beilagen usw., Arbeitsziele.

Laboraufbau im E502:



Aufgaben:

- Einbinden der FreeS/WAN IPsec Erweiterungen in einen SuSE Linux 2.2.x Kernel.
- Austesten des manuellen und automatischen Schlüsselaustauschs (IKE) zwischen den beiden Security Gateways .
- Verifizierung der korrekten Authentisierung und Verschlüsselung des IPsec Tunnels mittels tcpdump .
- Austesten der Road Warrior Funktionalität.
- Public Key Authentisierung der Verbindungspartner auf der Basis von X.509 Zertifikaten
- Eventuell Einbinden von zusätzlichen Verschlüsselungsalgorithmen, wie IDEA, Blowfish und CAST.
- Ausführliche Dokumentation aller notwendigen Konfigurationsschritte.



4 Einleitung

4.1 Einführung

Bis zum Jahr 1993 war das Internet ein reines Forschungsnetz. Im Vordergrund stand die allzeit ausfallsichere Möglichkeit, zwischen den Hochschulen zu kommunizieren. Das ist auch nach wie vor erwünscht. Seit die Unternehmen das Internet als 'das' Medium für globale Kommunikation, Datentransfer und Handel, das im speziellen die finanzielle Transaktionen beinhaltet, entdeckt haben, ist der Bedarf nach Sicherheit im Internet stark angestiegen. Der globale Handel mit all seinen Facetten gilt als Motor der Informationsgesellschaft. Das weltweite Handelsvolumen betrug nach Untersuchungen von Forrest Research Ende 1998 rund 1.8 Mrd. US-Dollar. Für die folgenden Jahre wird jeweils mit einer Verdreifachung des Volumens gerechnet.

Man muss wissen, dass das Internet zu den öffentlichen Netzen zählt, wie das Telefonnetz, und es ist anzunehmen, dass jeder aus unzähligen Büchern und Spielfilmen mitbekommen hat, dass ein öffentliches Netz verschiedene Gefahren birgt. Ein klassischer Missbrauch des öffentliches Netzes ist nach wie vor die Wirtschaftsspionage. Um diesem Umstand entgegenzutreten haben verschiedene Internet-Produkte-Hersteller (Cisco, Compaq) auf die grosse Nachfrage nach Sicherheit im Internet reagiert und bieten auf diesem Gebiet massgeschneiderte Sicherheitslösungen an. Dabei haben sie einerseits die nötigen Sicherheitskomponenten für eine sichere Kommunikation übers Internet in ihren vorhandenen Produkte zusätzlich implementiert oder haben speziell für diesen Zweck ein Produkt entwickelt. Da die Sicherheitslösungen der Hersteller untereinander nicht kompatibel sind, hat die Internet Society ihrer Internet Engineering Task Force IETF den Auftrag gegeben, ein Sicherheits-Protokoll für das Internet zu entwickeln, so dass die Hersteller darauf aufbauen können und die Anwender nicht mehr auf eine herstellerspezifische Lösung angewiesen sind.

Der folgende Vergleich macht die Sicherheitsstufen der zurzeit gängigen verschiedenen Internet Protokolle deutlich:

Datenübertragungs - Protokolle	**Postversand**
Internet Protokoll IP	Postkarte
Pretty Good Privacy *PGP* (Schutz von EMails)	Brief in Couvert
Secure Socket Layer *SSL* (Schutz von WWW-Netzverkehr)	Kurierdienst
IP Security Protocol IPSec (Schutz von Internet Protokoll)	Überwachter Geldtransporter

Wie man in der obigen Analogie sieht, gewährleistet das Internet-Protokoll keinen sicheren Datentransfer. Es garantiert weder die Authentizität von Absender noch garantiert es die Vertraulichkeit der übermittelten Daten. Um den Unterschied zwischen Authentisierung und Verschlüsselung und deren Bedeutung zu zeigen, können Vergleiche aus unserem täglichen Leben gemacht werden. Die folgende Analogie bezieht sich auf ein Gespräch, das wir mit jemanden führen wollen. Die Authentisierung steht im Zusammenhang mit der Identität meines Gesprächspartners. Kenne ich ihn? Kann ich ihm vertrauen? Die Verschlüsselung ist die Analogie zum Ort, wo das Gespräch geführt wird. Auf einem öffentlichen Platz können andere mithören und zu Hause ist man ungestört.

Datenkommunikation		Gespräch	
Authentisierung	Verschlüsselung	Mit wem?	Wo?
Nein	Nein	Jemand Fremder	In der Öffentlichkeit
Ja	Nein	Ein Freund	In der Öffentlichkeit
Nein	Ja	Jemand Fremder	Zu Hause
Ja	Ja	Ein Freund	Zu Hause

Damit nun einzelne Personen oder Organisationen mit Aussendienstmitarbeitern (Mobile Clients), Tele-Heimworkern und Satelittenbüros übers Internet vertraulich kommunizieren können, verwendet man ein 'Virtual Private Network' (VPN). Mit dem Einsatz eines VPN vergrössert man die Reichweite eines lokalen Firmennetzes enorm, denn durch Anwendung verschiedener Techniken des VPN werden Benutzer übers Internet in das Firmennetz der Organisation so eingebunden, als ob sie sich im gleichen Netz befinden. Im wesentlichen werden in einem VPN folgende Verfahren angewendet:

- Authentisierung
- Verschlüsselung
- Tunneling
- Firewalls

Das Prinzip ist sehr einfach: Ein virtuelles privates Netz sichert die Kommunikation übers Internet vor unberechtigten Lauschangriffen. Dabei handelt es sich um eine IP-Verbindung, die Informationen übers Internet kodiert und sich nur autorisierten Benutzern als Klartext offenbart.

4.2 Hinweise zur Benützung dieser Dokumentation

Um die Kernaussagen der recht komplexen Thematik so verständlich wie möglich zu machen, wurden einige verwendete wichtige Begriffe im Zusammenhang mit VPN in einem alphabetisch geordneten Glossar beschrieben. Dadurch vereinfacht sich auch die Berichtstruktur, dessen Aufbau unser schrittweises Vorgehen widerspiegelt. Es wurde grosser Wert auf die Rekonstruktion der ausgeführten Arbeiten gelegt. In der Regel beginnen die vorgenommenen Arbeiten in einer allgemeinen Abhandlung und enden dann mit konkreten praktischen Lösungen.

Folgende Vereinbarungen gelten zusammengefasst für diese Dokumentation:

Verweis auf Glossar	Times New Roman 12 / kursiv	*E-Commerce*
Verweis auf Dateien, Scripte	Courier 10 / kursiv	*`Ipsec.conf`*
Eingabe von Befehlszeilen	Courier 10/ fett	**`man Ipsec.conf`**
Eingabe in Konfigurationsdateien	Courier 10 / fett	`ipsec0=`**`eth0`**
Verweis auf Quelle	[Nummer]	[4]
Verweis auf Kapitel	[→ Kapitelangabe]	[→ 3.1 IPsec]

4.3 Glossar

Brute-Force-Angriffe

Der Angreifer ist durch Abhören im Besitz der verschlüsselten Nachricht. Ausserdem kennt er den Chiffrieralgorithmus. Nur den Schlüssel zum Entchiffrieren besitzt er nicht. Jetzt probiert der Angreifer alle möglichen Schlüsselwerte aus, bis er den richtigen findet.

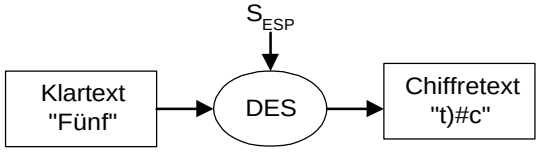
Beispiele von Brute-Force-Angriffe

Schlüsselart	Anzahl Bits	Anzahl Schlüssel	Testzeit eines Schlüssels	Anzahl Paralleltests	Mittlere Suchzeit
Kofferschloss mit drei Ziffern	10	1000	2s	1	17 Minuten
DES-Schlüssel	56	$\approx 2^{56}$	50us	1	57280 Jahre
DES-Schlüssel	56	$\approx 2^{56}$	0.02us	57600	3.5 Stunden

Wie man sieht, ist die ‘Mittlere Suchzeit‘ im wesentlichen abhängig von der Länge des benützten Schlüssels und der aufgewendeten Rechenleistung. Eine DES Verschlüsselung mit einem 56 Bit Schlüssel lässt sich mit den nötigen Mitteln, welche Regierungen oder grosse Firmen haben, in kürzester Zeit knacken. Aus diesem Grund wird das DES als nicht mehr Sicher betrachtet.

DES (Data Encryption Standard)

DES gehört zu den *Secret-Key-Systemen*. Es wurde 1977 veröffentlicht und verwendet einen Verschlüsselungsalgorithmus mit dem Namen ‘Lucifer‘, welche von IBM entwickelt worden ist. Ein 64 Bit offenen Text (Klartext) wird in 64 Bit verschlüsselten Text überführt unter Benutzung eines 56 Bit langen Schlüssels S_{ESP}. DES ist nicht zu empfehlen, da es mit einem Brute-Force-Angriff in kürzester Zeit geknackt werden kann. Aus diesem Grund verwendet man jetzt Triple-DES (*3DES)*.



E-Commerce

Elektronischer Handel spielt sich im Internet ab. Virtuelle Ladengeschäfte zum Durchstöbern oder elektronische Kataloge mit Suchfunktionen stehen zur Verfügung und bieten ihre Produkte an. Die Bestellung erfolgt direkt vom Terminal aus, die Bezahlung erfolgt rein elektronisch über Kreditkarten.

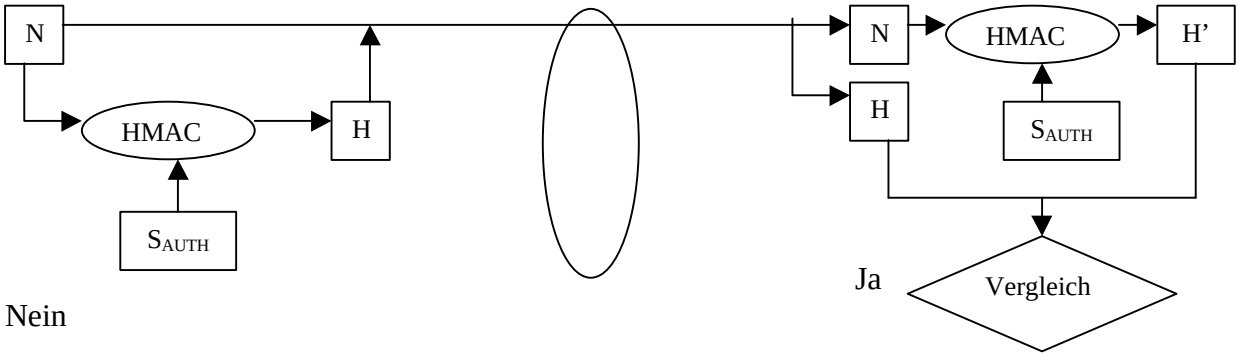
Hashalgorithmen / Hashwert

Diese werden auch Message Digests, Einweg-Hashfunktionen oder Prüfsumme genannt. Sie berechnen nach der Vorschrift H=h(N) aus einer beliebig langen Nachricht N einen Hashwert H fester Länge. Ein Hashalgorithmus hat folgende Eigenschaften:

- Einwegfunktion, weil es praktisch unmöglich ist, zu einem gegebenen Hashwert H die dazugehörige Nachricht N zu finden
- Bei einer Änderung nur eines Bits in der Nachricht N wird ein total anderer Hashwert berechnet.

Hashalgorithmen gibt es mit und ohne Schlüssel S_{AUTH}. Ein Hashalgorithmus ohne Schlüssel S wird verwendet um die Integrität der Nachricht zu überprüfen. Somit kann man überprüfen, ob die Nachricht in irgend einer Weise verändert wurde. Ein Hashalgorithmus mit Schlüssel S dient einerseits, wie oben erklärt, zur Datenintegritäts-Überprüfung und anderseits zur Authentifizierung. Mit Authentifizierung schaut man, ob man tatsächlich mit dem gewünschten Kommunikationspartner kommuniziert.

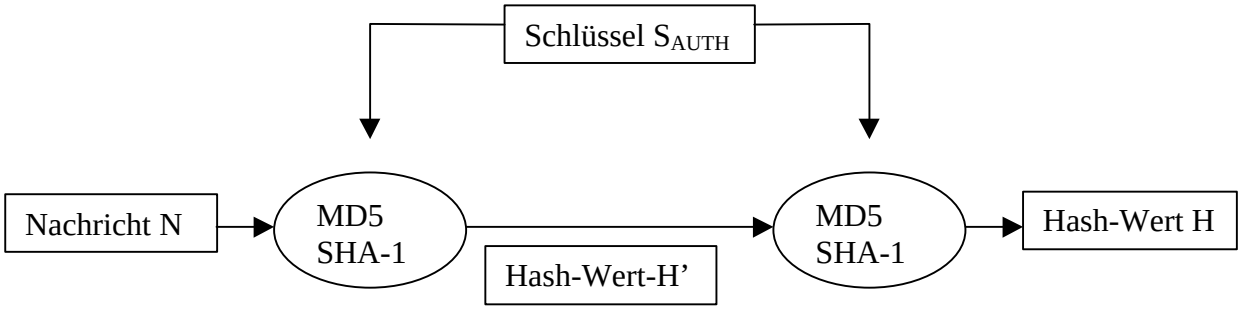
Die Authentifizierung-Prozess läuft folgendermassen ab:



57	Vermittlungskanal	Empfängerseite
Sender erzeugt einen H von der zu sendenden N	N und H werden über einen unsicheren Kanal zum Empfänger geschickt. Bsp. Internet	Empfänger erzeugt seinerseits einen Hashwert H' mittels N. Danach vergleicht er den Hashwert H des Senders mit seinem Hashwert H'. Falls das Resultat des Vergleichs übereinstimmt, dann kann man davon ausgehen, dass es sich um die gewünschte Person handelt. Denn nur Sender und Empfänger kennen den Schlüssel S. Gleichzeitig ist dadurch auch die Integrität der Daten gewährleistet. Falls das Resultat nicht übereinstimmt, wird die Nachricht N verworfen.

HMAC-MD5-96 or HMAC-SHA-1-96 (Hashed Message Authentication Code)

Dies ist eine spezielle Variante von *MD5* und *SHA-1*, die ein noch sicheres Verfahren darstellt. Beide, Sender und Empfänger, sind im Besitz der gleichen Nachricht N und des gemeinsamen Schlüssels S_{AUTH}. Bei diesem Verfahren wird je nach verwendetem Verfahren der *Hashalgorithmus* zweimal angewandt. Dabei ist zu beachten, das vom berechneten Hashwert H nur die 96 höchstwertigsten Bits verwendet werden. Das Vorgehen ist weiter unten illustriert.



ICMP – Internet Control Message Protocol

Im Internet werden mittels Router Datenpakete von einem Ort A zu einem Ort B transportiert. Falls während des Transportes etwas unerwartetes passiert, wird das Ereignis vom ICMP berichtet.
Ein ICMP-Nachrichtentyp ist immer in einem IP-Paket verkapselt. Diese Nachrichten können einerseits einem Systemverwalter wertvolle Hinweise für Fehlerursachen geben und anderseits zum Test von Verbindungen. Einige ICMP-Typen sollen hier kurz vorgestellt werden.

Typnummer im ICMP-Header	Nachrichentyp	Beschreibung
08	Echo Request	Fragt eine Maschine ob sie am Leben ist
00	Echo Reply	Maschine Antwortet, dass sie noch am Leben ist
03	Destination Unreachable	Paket kann nicht zugestellt werden

LDAP - Lightweight Directory Access Protocol (LDAP)

LDAP ist ein TCP /IP -basiertes Directory -Zugangsprotokoll, das sich im Internet und in Intranets als Standardlösung etabliert hat. Es ist abgeleitet vom X.500 Directory Access Protokoll (DAP), und ermöglicht den einfachen Zugang zu Directory-Systemen auf Basis von X.500, ebenso wie auf Nicht-X.500-Directories. Das LDAP-Protokoll definiert keinen Directory-Inhalt und auch nicht wie der Directory Service erbracht werden soll. Es setzt direkt auf TCP/IP auf und arbeitet auf Client-Server–Basis. LDAP hat ein weltweit eindeutiges Format in dem alle Namen darstellbar sind, es bietet unterschiedliche Layouts und eine eindeutige Zuordnung zwischen Namen und ihrer internen Repräsentation. Das Protokoll wird 1999 durch die IETF standardisiert.

MD5 (Message Digest)

siehe *Hashalgorithmen*

Private IP-Adressen

IANA (Internet Assigned Numbers Authority) hat einige Adressblöcke für den Einsatz als nicht registrierte IP-Adressen in privaten Netzen reserviert. ‘Registriert’ bedeutet, dass der Adressblock von einer Registrierungstelle für IP-Adressen stammt. Damit ist sichergestellt, dass es sich um Adressen handelt, die im Internet eindeutig sind. Da die gleichen privaten Adressen in mehreren lokalen Netzen vorkommen, ist die Eindeutigkeit nicht gegeben. Aus diesem Grund werden private IP-Adressen im Internet auch nicht geroutet. Damit diese Adressen trotzdem geroutet werden können, benützt man entweder einen Router mit NAT (Network Address Translation) oder ‘Tunneling’. Das NAT macht nicht anderes als eine Adressumsetzung, d.h., dass die Source-Adresse im IP-Header wird umgewandelt, und zwar wird die private IP-Adresse (Bsp. 10.0.2.1) durch eine offizielle IP-Adresse (Bsp. 160.85.131.61) ersetzt. Dadurch lässt sich jetzt dass Datenpaket im Internet routen. Das ‘Tunneling‘ löst dieses Problem, indem es das ganze Datenpaket mit privaten IP-Adressen in ein neues Datenpaket verpackt, dessen IP-Header offizielle Adressen besitzt.

Je nach der Grösse des lokalen Netzes benützt man die folgenden privaten IP-Adressen:

10.0.0.0	→	10.255.255.255	(Class A-Netz)
172.16.0.0	→	172.31.255.255	(Class B-Netz)
192.168.0.0	→	192.168.255.255	(Class C-Netz)

Public-Key-Systeme

Diese Verschlüsselungsverfahren werden auch als asymmetrische Verschlüsselungen bezeichnet, weil sie zur Ver- und Entschlüsselung nicht den gleichen Schlüssel benutzen. Sie benutzen einen privaten und einen öffentlichen Schlüssel. Der private Schlüssel muss geheim gehalten werden und wird beim Sender für das Verschlüsseln der Daten verwendet. Mit dem öffentliche Schlüssel, welcher der Empfänger besitzt, werden dann die Daten entschlüsselt.

Anwendung:	Schlüsselaustausch
	Authentifizierung ohne Versschlüsselung
	Digitale Signaturen
Algorithmen:	DH (Diffie-Hellman)
	RSA (Rivest, Shamir, Adleman)

Perfect Forward Secrecy

Je häufiger ein neuer Schlüssel für die Verschlüsselung benutzt wird, umso höher ist der Sicherheitsgrad. Damit keine Abhängigkeit zwischen dem alten und neuen Schlüssel entstehen, wird auch im Quick Modus (2. Phase des IKE-Protokolls) noch einmal mittels Diffie-Hellman ein geheimer Schlüssel erzeugt, mit deren Hilfe man dann die neuen Schlüssel berechnet.

PGP Pretty Good Privacy

PGP ist ein Protokoll zum Versenden von sicheren E-Mails. Es bietet die Möglichkeit, öffentliche Schlüssel durch Zertifikate gegen Man-in-the-Middle-Angriffe zu schützen. Zur Verschlüsselung werden die beiden Verfahren RSA und IDEA angewendet. PGP ist im Internet weit verbreitet, weil es als Public Domain für viele Plattformen verfügbar ist.

Security Association SA

Das ist ein 'Vertrag' über Sicherheitsparameter einer Kommunikationsbeziehung wie Verschlüsselungsverfahren, Authentfizierungsverfahren, Schlüsselmaterial, Gültigkeitsdauer des Schlüsselmaterials usw. Anhand dieser SA's werden dann die gesendeten Datenpaket bearbeitet.

Secret-Key-Systeme

Secret-Key-System bezeichnet man auch als symmetrische Verschlüsselungen. Diese Verschlüsselungs-Algorithmen verwenden für die Verschlüsselung und Entschlüsselung den gleichen Schlüssel.

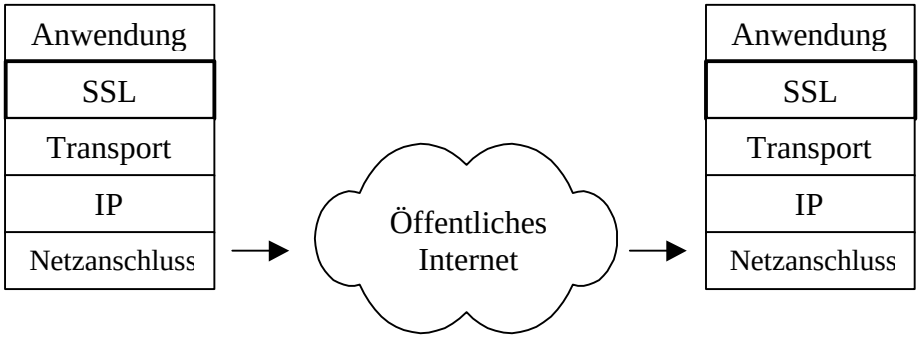
Algorithmen: DES (Data Encryption Standard)
IDEA (International Ata Encryption Algorithmus)

SHA-1 (Secure Hash Algorithm Nr.1)

siehe *Hashalgorithmen*

SSL

Secure Sockets Layer (SSL) ist ein Protokoll, das die Daten kryptographisch schützt und sie dann dem TCP/IP-Stack übergibt. SSL enthält die drei gebündelten Protokolleigenschaften Authentifizierung, Verschlüsselung und Schlüsselaustausch. SSL ist typischerweise in eine Anwendung integriert (z.B. Browser). Es wurde von Netscape Communications im Rahmen ihres Sicherheitspakets für das World Wide Web entwickelt.



TCP/IP- Referenzmodell (Transmission Control Protocol/Internet Protocol)

TCP/IP wurde vom amerikanischem Verteidigungsministerium für das ARPANET (Advanced Research Project Agency Network) entwickelt. Weil das Internet auf diesem Protokoll aufbaut, hat es heute grosse Bedeutung für Computernetzwerke. Um die Funktionsweise dieses Protokolls theoretisch zu verstehen, hat man TCP/IP in 5 Schichten, welche Protokolle repräsentieren, aufgeteilt. Jede Schicht stellt der jeweils darüberliegenden Schicht Dienste bereit.

TCP/IP-Referenzmodell

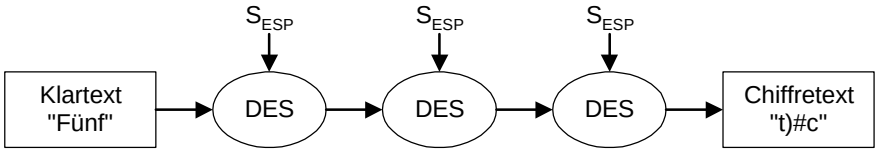
Schicht 5	Anwendungsschicht (eigentliche Anwendung ,WWW, Telnet, usw.)
Schicht 4	Transportschicht (verbindungsorientert oder verbindungsose Verbindung)
Schicht 3	Vermittlungsschicht (festlegen des Transportweges)
Schicht 2	Sicherungsschicht (einfache Fehlerkorrekturen)
Schicht 1	Bitübertragungsschicht, (Bits auf Kupfer, Glas, etc.)

X.509

X.509 ist ein Standart der ITU (International Telegraph Union) für Zertifikate, welche in Public-Key-Systemen eingesetzt werden. Er spezifiziert das Format des Zertifikates sowie die Bedingungen für die Herstellung und Benutzung.

3Des (Triple Data Encryption Standard)

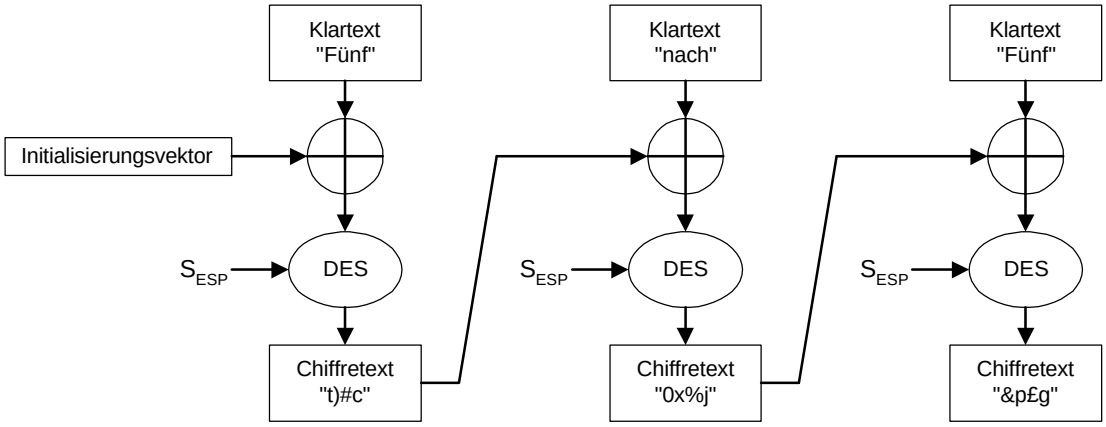
Triple-DES ist ein Verfahren, das den *DES*-Algorithmus dreimal auf den Klartext anwendet.



Dadurch wird die Schlüssellänge 3*56Bit=168 Bit lang und somit für einen Brute-Force-Angriff unknackbar. Das bedeutet aber auch, dass der Aufwand (Rechenleistung, Zeit) für das Ver- und Entschlüsseln dreimal so hoch ist.

3Des-CBC (Triple Data Encryption Standard-Cipher Block Chaining)

3DES hat einen entscheidenden Nachteil. Bei Eingabe des gleichen Textes wird der gleiche Chiffretext erzeugt. Dies gilt es zu verhindern, denn durch Wiederholungen im Chiffretext können Rückschlüsse auf den Inhalt des Klartextes gezogen werden. Das ist somit ein Schlupfloch für Codebrecher. Aufgrund dessen wurden verschiedene Modi entwickelt, um wiederholt auftretende Klartextblöcke zu verschleiern. Im FreeS/WAN wird der 3DES-CBC Mode angewandt.



Im CBC-Mode wird der erste Klartextblock vor der Verschlüsselung mit einem Initialisierungsvektor und jeder andere Klartextblock mit dem jeweils vorangehenden Chiffretextblock XOR-verknüpft. Im Gegensatz zum *3DES* werden somit wiederholt auftretende Klartextsequenzen bei jeder Verschlüsselung in einen anderen Chiffretext überführt. Dadurch ist die Verschlüsselung schwieriger zu knacken.

5 VPN – Theorie

5.1 Was ist ein ‘Virtual Private Network‘?

Ein virtuelles privates Netz verbindet lokale Netze (Intranetze) über öffentliche Netze (Telefonnetz, Internet) miteinander. Der Ausdruck “privat” bedeutet, dass die Verbindung zwischen Rechnern genauso gut gesichert ist, wie wenn die Rechner zusammen in einem geschützten lokalen Netz stehen würden. Obwohl die Rechner räumlich durch das öffentliche Netz getrennt sind, hat man durch das Tunneling-Verfahren eine Situation geschaffen, welche die Rechner ‘virtuell’ wie in einem lokalen Netz verbindet. Angesichts dieser Gründe folgte der Name ‘**Virtual Private Network**’.
Wie erwähnt, werden die Netze über ein unsicheres öffentliches Netz verbunden, trotzdem sind durch VPN folgende Sicherheitsvoraussetzungen für eine gesicherte Verbindung gegeben:

- Authentifizierung des Kommunikationspartners
- Integrität der Informationen, d.h., die gesendeten und empfangenen Daten sind nicht verändert worden
- Abhörsicherheit durch Verschlüsselung
- Identitätsverbergung der Kommunikationspartner und Protokollunabhängigkeit ab der dritten Schicht durch Tunneling
- Schutz des lokalen Netzes vor dem öffentlichen Netz durch einen Firewall

Das Bilden von privaten Netzen ist nicht neu. Die alte Methode verwendet dafür temporäre oder permanent gemietete öffentliche Leitungen vom Telefonnetz , welche somit privat und daher als ‘sicher’ betrachtet werden kann. Die Skizze soll einen Überblick über die alte Methode des Anschlusses an das Firmennetz illustrieren.

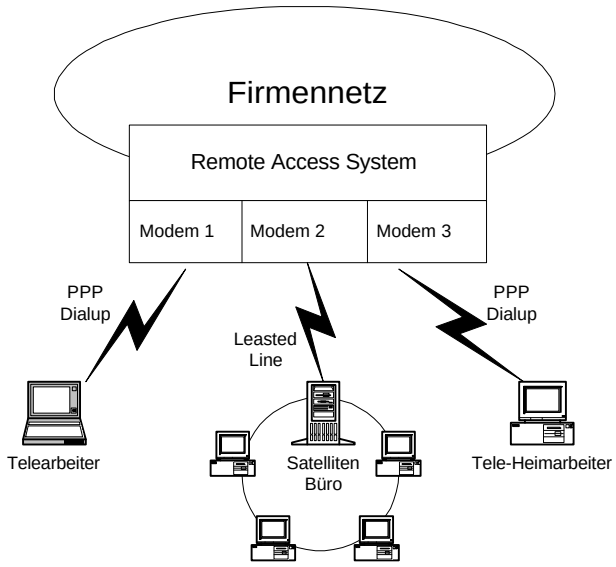


Abb. 5.1.1

Mobile Client und Tele-Heimarbeiter wählen sich mittels eines Modems in das lokale Netz ein. Je nach Distanz (Ferngespräche) kann das zu sehr teuren Telefonkosten führen.

Verbindungen mittels Mietleitungen zu eigenen lokalen Satellitenbüronetzen mit dem Firmennetz sind trotz steigendem Konkurrenzkampf der Leitungsanbieter eine teure Angelegenheit. Die Gebühren einer Mietleitung beinhalten Anschluss und Distanz.

Das benützen des Internets als öffentliches Netz bringt den Unternehmen einige Vorteile:

Einsparungen von 60-80% der Telefonkosten bei Tele-Heimarbeitern und mobile Clients Ferngespräche bezahlt man jetzt unter Nutzung des Internets zum Ortstarif.
Einsparungen von 20-50% der Kosten von Mietleitungen.
Weltweite Zugriffsmäglichkeiten aufs Internet und somit auf das lokale Firmennetz.
Weniger Hardware und somit weniger Unterhaltsarbeit im Firmennetz, also kein Remote Access System mit seinen Modems mehr im Firmennetz.
Förderung des *E-Commerce* übers Internet

Die Skizze illustriert VPN-Lösungen übers Internet

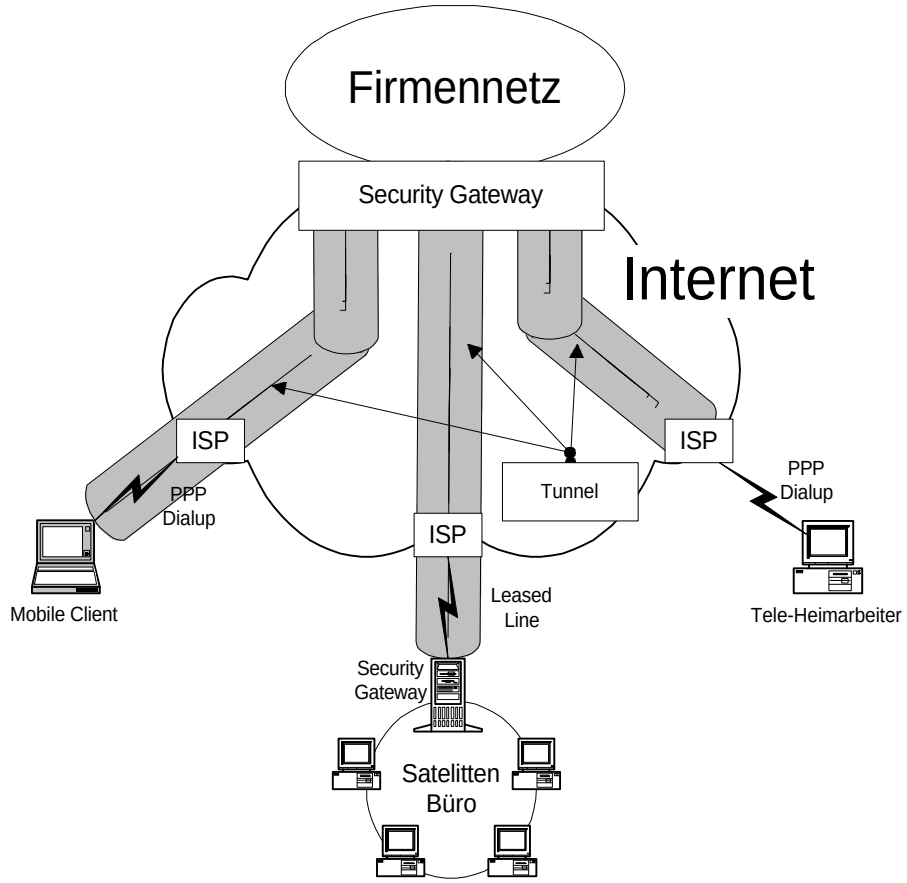


Abb. 5.1.2

Wie man sieht, gibt es verschiedene VPN Architekturen. Allen gemeinsam ist die Benützung eines Tunnels, das entweder bis zu den Host's selber geht oder nur bis zum ISP (Internet Service Provider). Für den kommerziellen Bereich gibt es drei exemplarische Fälle für den Einsatz von Virtual Private Networks.

5.2 VPN Architekturen

5.2.1 End-to-End

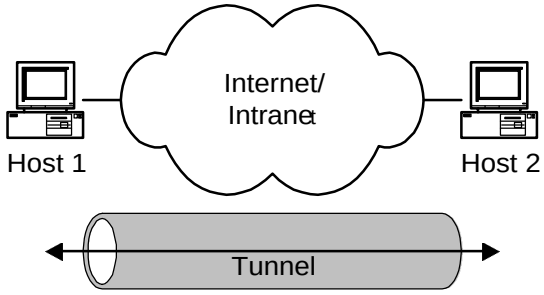


Abb. 5.2.1.1

Diese ist die sicherste Lösung für ein VPN-Verbindung über das Internet. Der Tunnel mit den verschlüsselten Daten deckt die ganze Verbindung bis zu den Hosts ab. Dadurch kann eine Angriff auf der ganzen Verbindungslänge ausgeschlossen werden. Dazu muss jeder an der verschlüsselten Kommunikation beteiligter Host mit entsprechender VPN-Software ausgestattet sein. Voraussetzung ist aber, dass die Host-Rechner leistungsfähig sind, damit der Aufwand und die Verzögerung, welche die VPN-Software naturgemäss mit sich bringt, im Rahmen bleiben.

5.2.2 Site-to-Stite

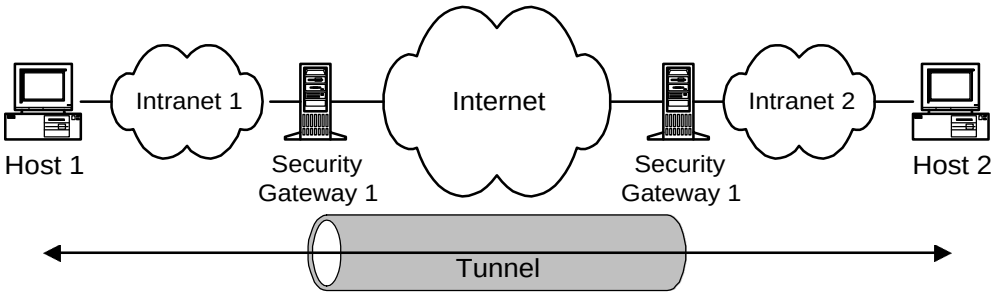


Abb. 5.2.2.1

Bei Site-to-Site tauschen zwei Intranetze mit ihren Stationen Daten übers Internet aus [→ Abb. 5.2.2.1]. Die Kommunikation über das Internet ist verschlüsselt und innerhalb eines Tunnels. Der Vorteil dieser Art der Verbindung von Rechnern über VPN's liegt darin, dass keine der lokalen Arbeitsstationen mit spezieller VPN-Software ausgerüstet sein muss. Da die Gateways die ganze Arbeit mit der Sicherheit erledigen, ist das VPN für die Rechner im LAN vollständig transparent. Die Verwendung von sehr leistungsfähigen Security Gateways wird vorausgesetzt. Neben der Belastung der Hosts senkt dies natürlich den zusätzlichen Verwaltungsaufwand für den Administrator durch ein VPN erheblich. Falls das Vertrauen gegenüber dem Serviceprovider vorhanden ist, kann der ganze Sicherheit-Aufwand dem ISP Internet Service Provider überlassen werden. Damit überträgt man den administrativen Aufwand und den Support dem Provider. Aus sicherheitstechnischen Gründen ist das sicher nicht eine optimale Lösung.

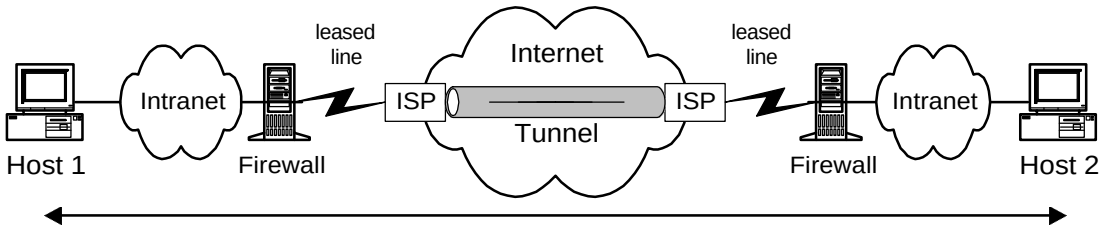


Abb. 5.2.2.2 VPN-Lösung wird Service-Provider angeboten.

5.2.3 End-to-Site

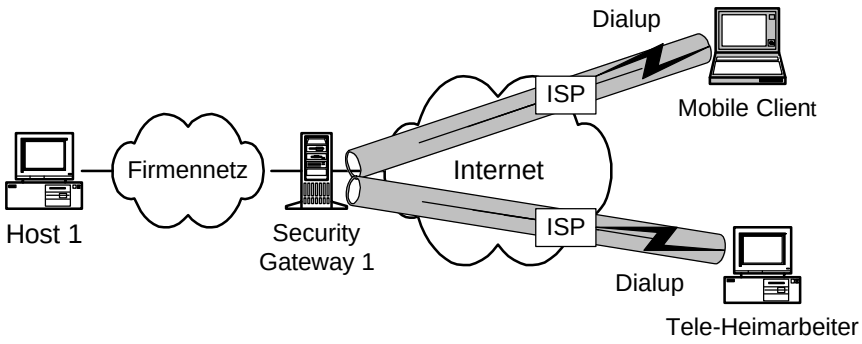


Abb. 5.2.3.1

Bei der End-to-Site Kommunikation handelt es sich um eine Kombination der beiden vorangegangenen Fälle mit ihren Vor- und Nachteilen. Mit dieser Verbindungsart werden die mobilen Clients und Tele-Heimarbeiter ins VPN miteinbezogen. Dadurch lassen sich die vorher schon erwähnten Einsparungen bei den Telefonrechnungen erzielen, so dass in kurzer Zeit die Kosten für das VPN-Produkt wieder hereingeholt ist. Wie man sieht wählen sich die Benützer bei ihrem ISP ein und bauen dann eine sichere Verbindung zum Firmennetz auf.

5.3 Was für VPN-Protokolle gibt es?

Da das ursprüngliche *TCP/IP–Referenzprotokoll* des Internets keinen Sicherheitsaspekt bietet, musste das *TCP/IP-Referenzmodell* mit Sicherheitsprotokollen ergänzt werden. Auf dem Internet- Markt gibt es viele Protokolle, welche zur Realisierung eines VPN verwendet werden können. Die Protokolle können in die verschiedenen Schichten des *TCP/IP-Referenzmodell* eingeteilt werden.

TCP/IP-Referenzprotokoll	Sicherheits-Protokolle	Kurze Beschreibung
Applikationsschicht	**IPSec (IKE)** S-HTTP S-MIME	IP Internet Security (Internet Key Exchange) Secure Hyper Text Transfer Protocol Sichere Übertragung von WWW-Seiten Secure Multiprupose Internet Mail Extention Standard zur sicheren Übertragung von Email
TCP/UDP Transportschicht	SOCKS SSL	Socket Secure Server, Standard zur Nutzung von Internet-Diensten über einen Firewall Secure Sockets Layer,Netscapes Technik zur sicheren Üertragung von HTTP (Hyper Text Transfer Protocol)
IP Vermittlungsschicht	IPSec(AH,ESP) Paket filtering	**IP Internet Security** Firewall
Sicherungsschicht	L2TP PAP CHAP	Layer 2 Tunneling Protocol Password Authentication Protocol (PAP) Challenge Handshake Authentication Protocol
Bitübertragungsschicht		

In diesem Projekt kommt für die Realisierung des VPN das Sicherheits-Protokoll IPSec zur Verwendung. Wie man aus der Tabelle entnehmen kann, arbeitet IPSec mit AH, ESP und IKE.
Zu diesem Modell ist noch zu sagen, dass die Protokolle auf dem dritten Layer die universellsten sind, denn die in den höheren Layern schützen nur eine spezifische Anwendung, und die in den unteren Layern sind mediumspezifisch.

5.4 IPSec – IP Internet Security

Entwickelt und verwaltet wird dieser VPN-Standard von der IETF- Internet Engineering Task Force. Mit IPSec steht ein allgemein verbindlicher, herstellerübergreifender Standard zur Verfügung, der den Datenaustausch zwischen unterschiedlichen Security Gateways im Rahmen einer VPN-Lösung regelt. Die zu verwendeten Protokolle im Rahmen des IPSec-Standards müssen folgende Aufgaben bewerkstelligen:

- Authentifikation der Gesprächspartner
- Integrität der Informationen
- Verschlüsselung der Informationen
- Massnahmen gegen Replay-Angriffe
- Schlüssel Management

Um diese Forderungen zu erfüllen, verwendet man das AH- (Authentication Header), ESP- (Encapsulating Security Payload) und das IKE (Internet Key Exchange) Protokoll.

Bevor diese Protokolle näher betrachtet werden, sollen die zwei Modi vorgestellt werden, welche IPSec benützt. Je nachdem, ob man intern im lokalen Netzwerk kommuniziert oder extern über ein öffentliches Netz, hat man die Wahl zwischen Transportmodus und Tunnelmodus.

5.4.1 Transportmodus

Dieser Modus wird mehrheitlich innerhalb eines sicheren internen Netzes verwendet. Aus diesem Grund ist der angewendete Sicherheitsgrad geringer als im Tunnelmodus. Das ursprüngliche Datenpaket wird nur in soweit verändert, wie es nötig ist, um die Protokolle AH und ESP anzuwenden. Das bedeutet, dass der ursprüngliche IP-Header erhalten bleibt und je nach dem, ob AH oder ESP angewendet wird, sind die Daten entweder nur authentisiert oder authentisiert und verschlüsselt.
Das positive dieses Modus ist das Sparen von Rechenzeit, weil die Datenpakete weniger Rechenintensiv bearbeitet werden müssen. Dies kann zum Beispiel für Echtzeit-Anwendungen, wie Telefonieren über das Internet, von Bedeutung sein.

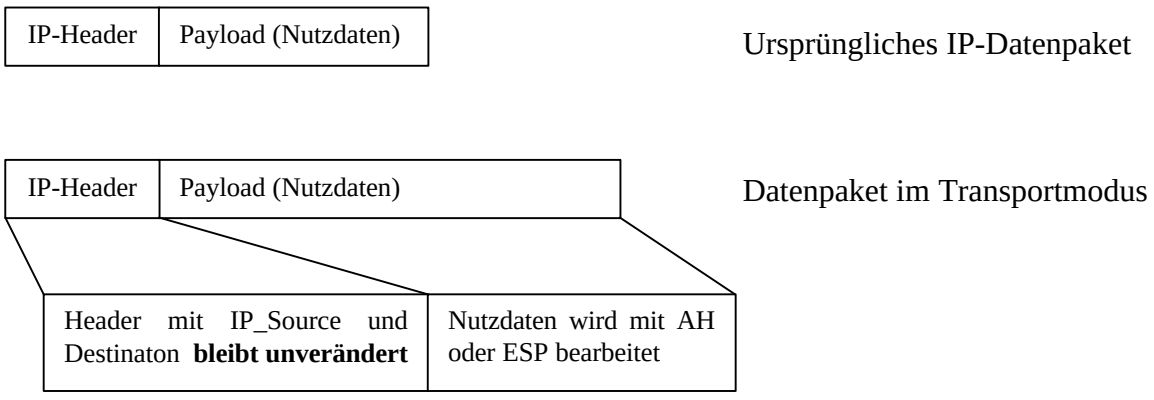


Abb. 5.4.1.1

5.4.2 Tunnelmodus

Dieser Modus wird für Verbindungen verwendet, welche über ein öffentliches Netz , wie das Internet geht. Der Tunnelmodus in Verbindung mit dem ESP ist dafür das geeignetste Mittel.
Die ursprüngliche Anwendung von Tunneling ist ein lokales Netz, welches zum Beispiel die Protokolle NetBIOS (IBM) und IPX (Novell) verwendet, um über ein TCP/IP-Netz zu übertragen. Dies wird erreicht, indem das originale Datenpaket in einen IP-Datenpaket 'eingepackt' über das TCP/IP-Netz übertragen wird. Somit beinhaltet das neue Datenpaket das urspüngliche Datenpaket als seine Nutzdaten.
Mit dem Ziel einen grösseren Sicherheitsgrad zu erreichen, wird diese Technik in IPSec angewendet, denn durch das 'Einpacken' und zusätzlichem Verschlüsseln mittels ESP wird das gesamte ursprüngliche Datenpaket verhüllt. Somit bleibt im Tunnelmodus die Identität der Source- und Destination-Adresse im Verborgenen, oder anders gesagt, die Identität der Kommunikationspartner bleibt anonym. Das ist ein Vorteil gegenüber dem Transportmodus.

Bezogen auf unser Modell sieht es folgendermassen aus:

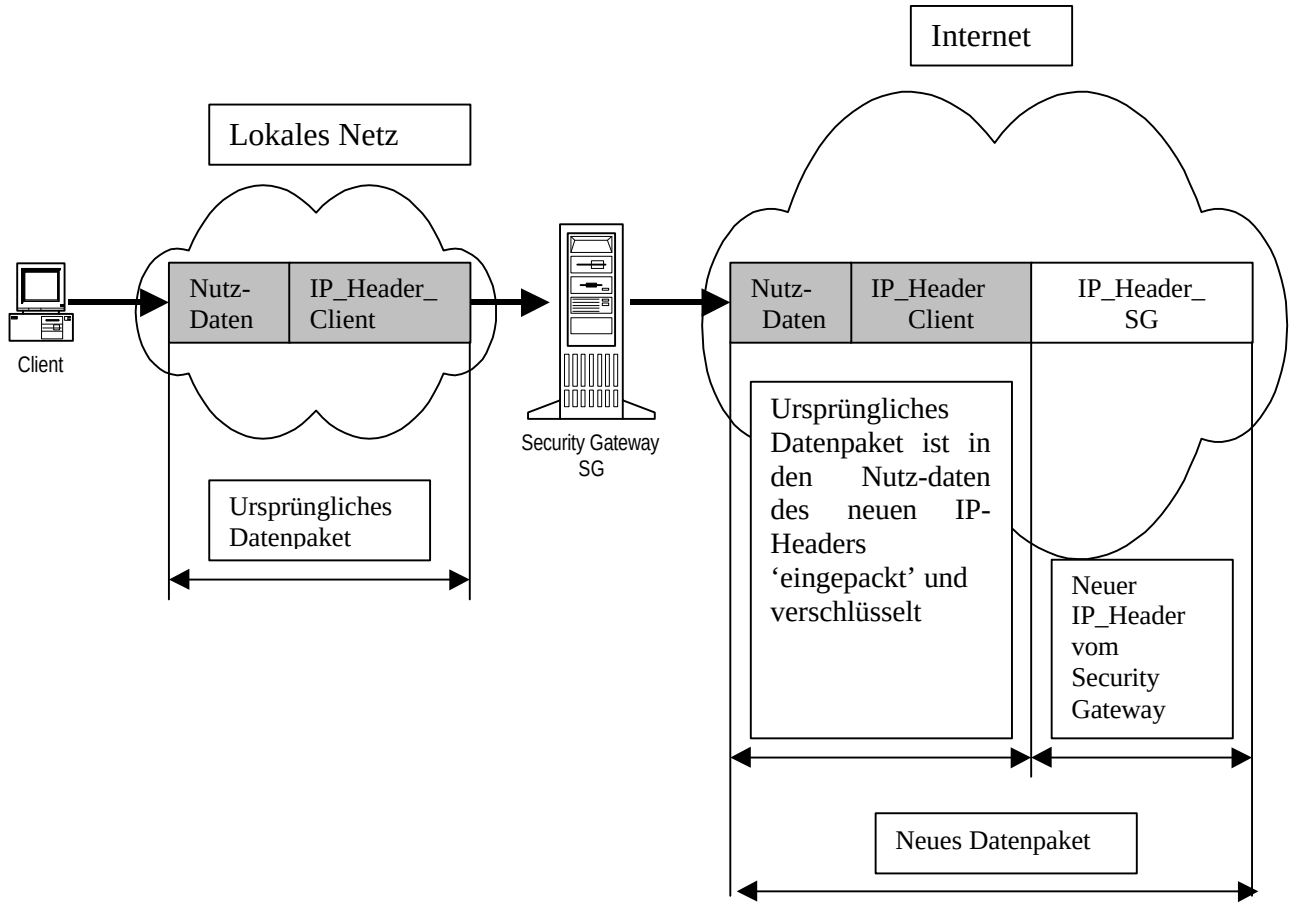


Abb. 5.4.2.1

Ein weiteres Plus von Tunneling ist die Benützung von *privaten IP-Adressen*. In der Regel ist das lokale Netz mit *privaten IP-Adressen* aufgebaut. Wie man in der Skizze sieht, wird der IP-Header_Client (Bsp. 10.0.1.2) in den Nutzdaten des neuen Datenpakets versteckt, welcher einen IP-Header_SG hat, der routingfähig und somit eine registrierte IP-Adresse (Bsp. 160.85.131.60) hat. Somit kann man also mit nicht routingfähigen Adressen, Datenpakete durch ein öffentliches TCP/IP-Netz senden.
Nachdem nun die zwei Modi erklärt worden sind, gehen wir nun auf die IPSec Protokolle AH, ESP und IKE näher ein.

5.5 Authentication Header (AH)

Das Authentication-Header-Protokoll (AH) erzeugt bei einem Datenpaket einen zusätzlichen Header. Dieser Header enthält die nötigen Informationen um eine Authentifikation durchzuführen. Eine Authentifikation deckt die folgenden drei Sicherheits-Anforderungen ab:

- Bestätigung, dass das empfangene Datenpaket vom richtigen Sender kommt
- Sicherstellen der Datenintegrität
- Schutz gegen Replay-Angriffe

Die ersten zwei Forderungen werden mittels eines *Hashwertes* überprüft, welcher durch einen *Hashalgorithmus* erzeugt worden ist. Es stehen zwei Hashalgorithmen zur Verfügung:

HMAC-MD5, erzeugt einen Hashwert mit 128 Bit Länge
HMAC-SHA, erzeugt einen Hashwert mit 160 Bit Länge

Der Replay-Angriff wird durch die Angabe der Folgenummer verhindert, den damit kann der Empfänger erkennen, ob ein Datenpaket wiederholt gesendet wurde. Die Skizze zeigt den Aufbau eines AH-Headers.

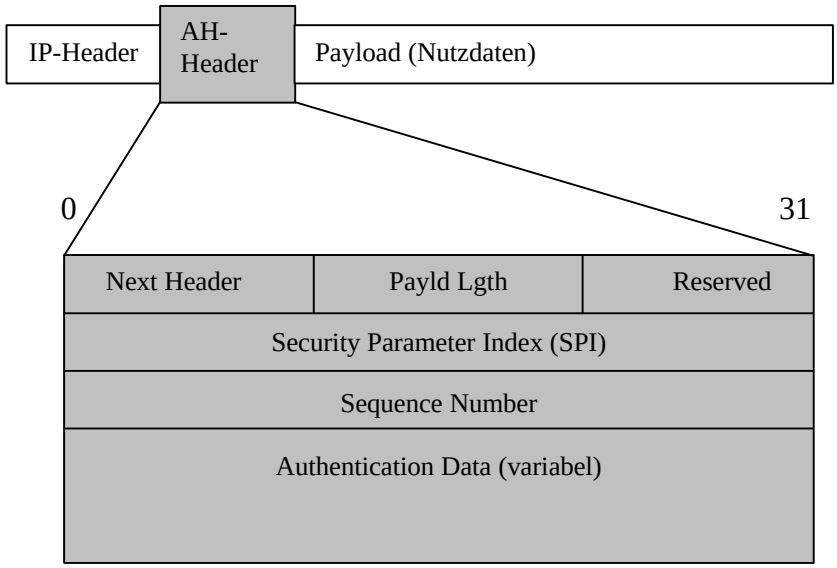


Abb. 5.5.1

Next Header:	Identifiziert den Typ des Payloads, also ob es sich zum Beispiel um ein TCP (Nr. 6) oder um ein UDP (Nr. 17) handelt.
Payload Length:	Länge des AH-Headers.
Reserved:	Für zukünftige Anwendungen reserviert.
Security Parameter Index SPI:	Dieser Wert ist ein Pointer, welcher auf die *Security Association SA* zeigt, welche für dieses Datenpaket zuständig ist.
Sequence Number:	Schutz gegen Reply-Angriffe.
Authentication Data:	Hashwert, je nach dem welcher Hashalgorithmus verwendet wurde ist dieser Eintrag verschieden lang.

Die Anwendung von AH bezogen auf die zwei Modi sieht folgendermassen aus:

AH im Transportmodus

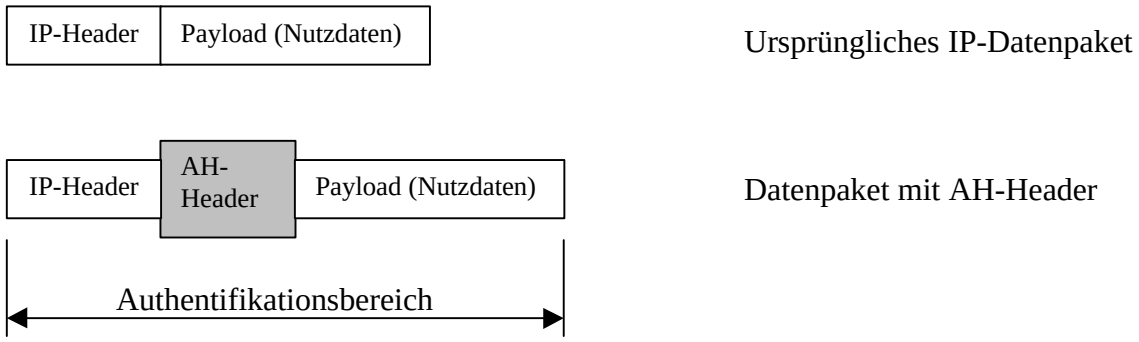


Abb. 5.5.1.1

AH im Tunnelmodus

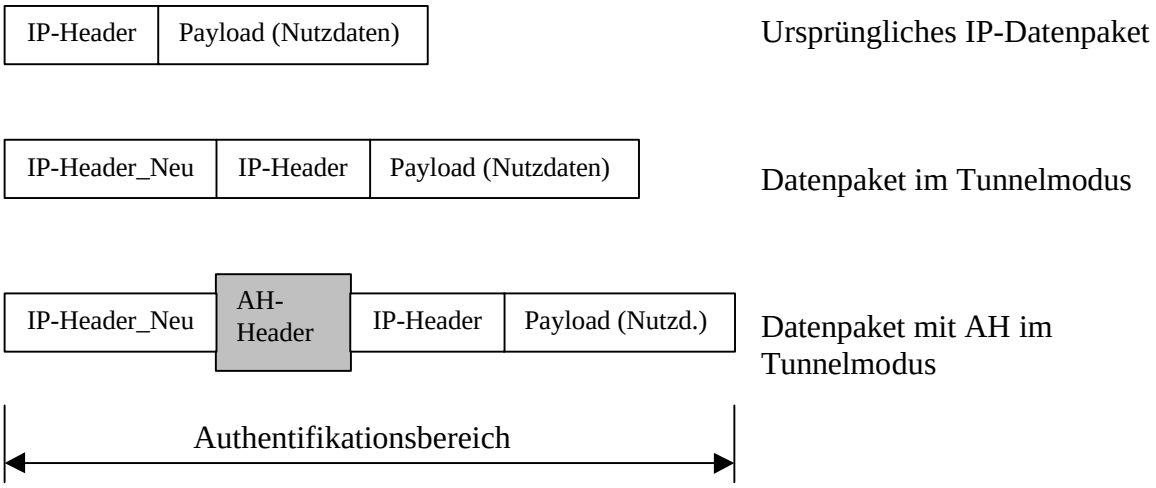


Abb. 5.5.1.2

Obwohl der AH-Header mittels des Hashwertes das ganze Datenpaket abdeckt, gibt es einige variable Felder (Bsp. Time to Live, Header Checksum, usw.) , die im Hashwert nicht berücksichtigt werden, da sie beim Transport vom Ursprungsort zum Zielort verändert werden und somit den Hashwert ungültig machen würden.

5.6 Encapsulating Security Payload (ESP)

Der Unterschied zum AH-Protokoll ist, dass bei ESP die Verschlüsselungskomponente dazukommt. Das heisst, bei diesem Verfahren werden vier Sicherheitsanforderungen erfüllt.

- Bestätigung, dass das empfangene Datenpaket vom richtigen Sender kommt
- Sicherstellen der Datenintegrität
- Schutz gegen Replay-Angriffe
- Vertraulichkeit der gesendeten Informationen

Durch das zusätzliche Verschlüsseln werden die Informationen vor unberechtigten Lesern geschützt. Folgende Verschlüsselungsalgorithmen können benutzt werden:

DES_CBC	(RFC2405)	Data Encryption Standard_ Cypher Block Chaining
IDEA	(RFC 2451)	International Data Encryption Standard
Blowfish	(RFC 2451)	
3DES	(RFC 2451)	Triple Data Encryption Standard
CAST_128	(RFC 2451)	

Auch mit ESP wird eine Authentifikation durchgeführt. Im Gegensatz zu AH wird aber nicht das ganze Datenpaket mit einem Hashwert abgedeckt. Im ESP bleiben die IP-Header unberücksichtigt.
Kommt ein mit ESP gesendetes Datenpaket beim Empfänger an, dann wird zuerst die Authentifikation durchgeführt. Falls diese in Ordnung ist, wird die Entschlüsselung eingeleitet. Mit diese Vorgehensweise soll der Prozessor mit der sehr rechenintensiven Entschlüsselung nur dann belastet werden, wenn es wirklich notwendig ist. Dadurch verringert sich die Verletzbarkeit des Computers gegen 'Denial of Service'-Attacken [→ Angriffe im Netz].
Die Skizze zeigt den Aufbau eines ESP-Datenpakets. Wie man sieht, sind durch den höheren Grad der Sicherheit auch der Umfang an Zusatzinformationen gestiegen.

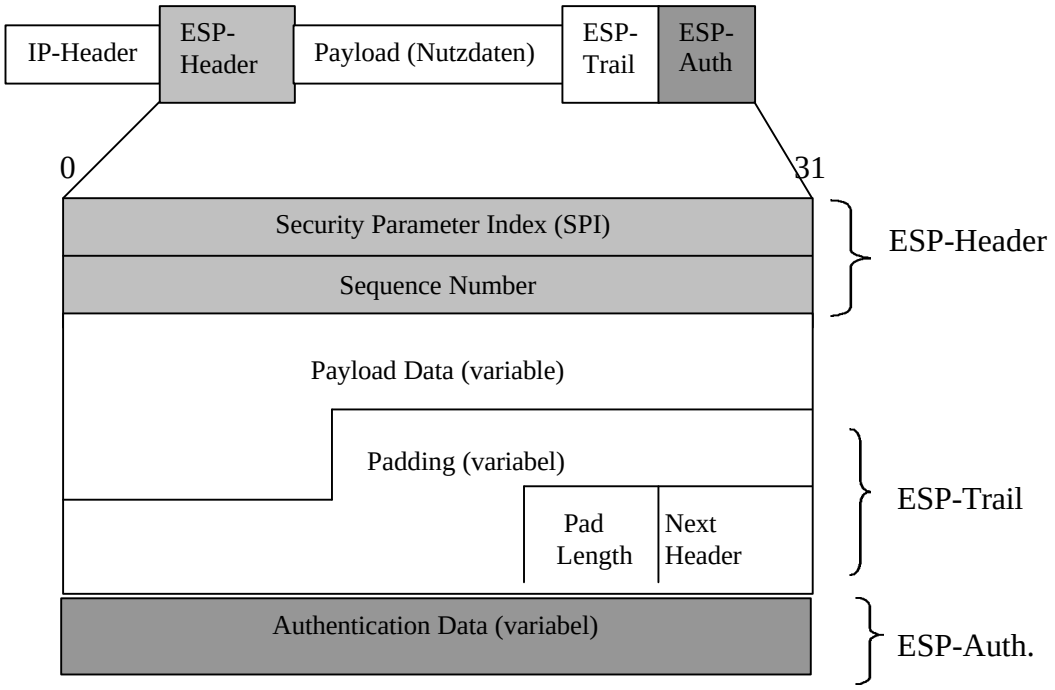


Abb. 5.6.1

Security Parameter Index SPI:	[→ Authentication Header AH]
Sequence Number:	[→ Authentication Header AH]
Payload Data:	Verschlüsselte Nutzdaten
Padding:	Je nach verwendetem Verschlüsselungsalgorithmus wird als Input eine ganz bestimmte Länge des Datenpakets verlangt um die Verschlüsselung durchzuführen. Das Padding dient zum Erreichen der gewünschten Länge.
Pad Length:	Länge des vorangehenden Padding Feldes.
Next Header:	Daten-Typ der Nutzdaten (TCP/UDP etc.)
Authentication Data:	Im ESP ist das Erzeugen eines Hashwertes optional, aber aus Sicherheitsgründen wird es in der Regel gemacht. Auch wenn die Daten verschlüsselt sind, wäre die Möglichkeit gegeben eine Fälschung der Daten vorzunehmen. Durch den Hashwert wird das ganz klar verunmöglicht.

Die Anwendung von ESP bezogen auf die zwei Modi sehen folgendermassen aus:

ESP im Transportmodus

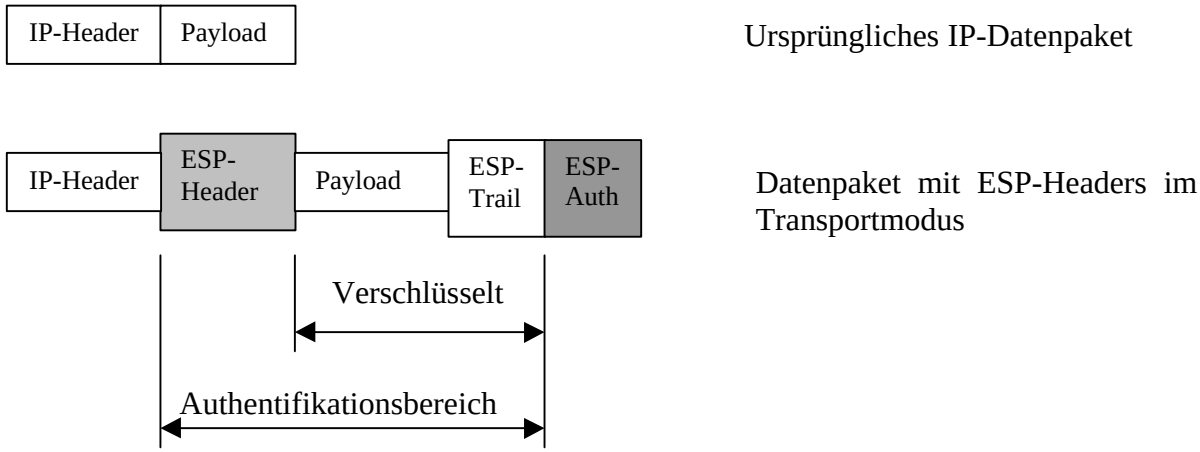


Abb. 5.6.1.1

ESP im Tunnelmodus

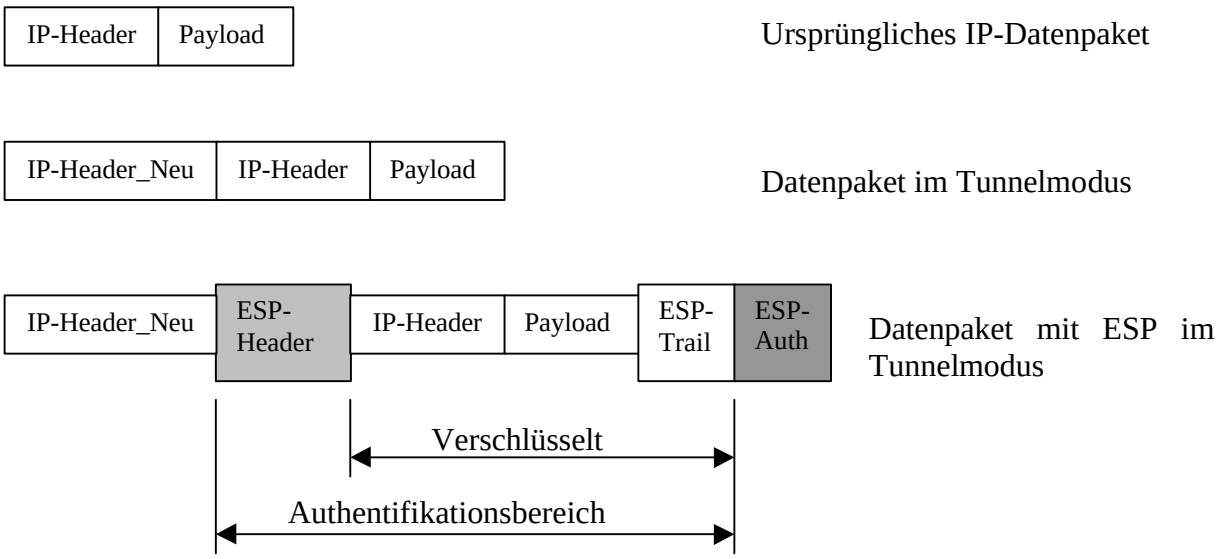


Abb. 5.6.1.2

Jetzt könnte man sich die Frage stellen, warum brauchen wir das AH-Protokoll. Es genügt doch, wenn wir im ESP-Protokoll zusätzlich zur Verschlüsselung einen AH-Header erzeugen. Zwei Gründe dafür sollen hier kurz erwähnt werden:

In bestimmten Ländern (Frankreich), wo Kryptographie verboten ist , hat man gar kein andere Wahl, als das AH-Protokoll zu verwenden. Somit gibt es also für das AH-Protokoll weltweit keinen Einschränkung für deren Benützung.
Eine Verbindung, welche nur Authentifikation braucht, zum Beispiel Kommunikation innerhalb eines firmeninternen, abhörsicheren Netzes, spart man Zeit (Prozessor muss weniger arbeiten) und Bandbreite (Datenpaket enthalten weniger zusätzliche Headers).

5.7 Praktisch genutzte Kombinationen

Wie wir gesehen haben, gibt es verschiedene Möglichkeiten eine sichere Verbindung aufzubauen.
Je nach Situation wählt man den geeigneten Modus und das passende IPSec-Protokoll aus. In der Praxis haben sich die folgenden Kombinationen als die sinnvollsten herausgestellt:

5.7.1 Host-to-Host-Verbindung

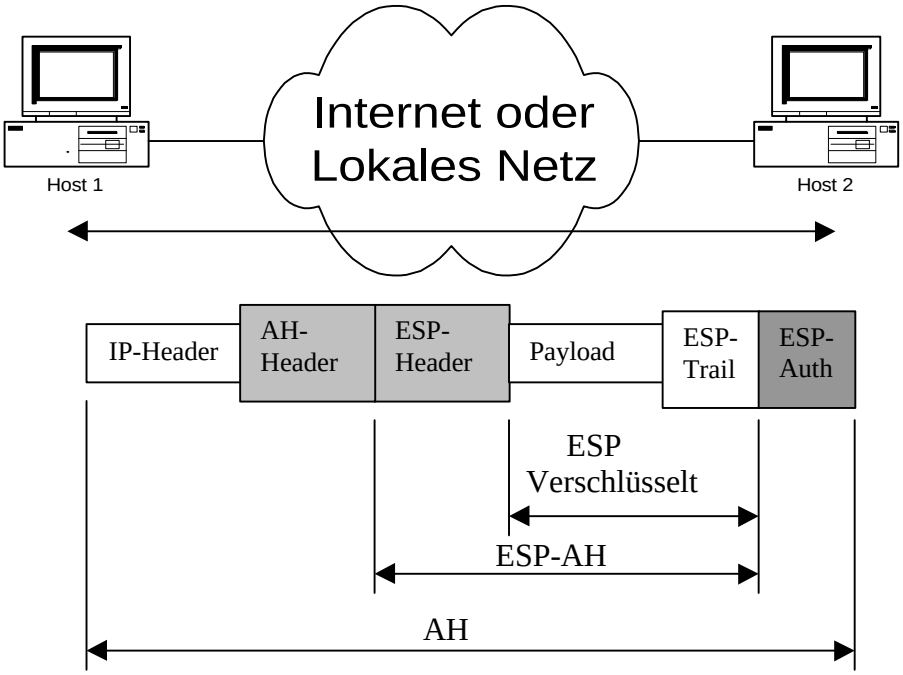


Abb. 5.7.1.1

Bei Host-to-Host-Verbindungen macht es keinen Sinn Tunneling zu verwenden, da der IP-Header der gleiche wäre. Darum benützt man den Transportmodus mit AH und ESP. Wie man sieht, wird mit dem zusätzlichen benützen des AH-Headers die Authentifikation auf das ganze Datenpaket ausgeweitet. Dadurch erhöht sich natürlich der Sicherheitsgrad der Verbindung.

5.7.2 Typische VPN-Verbindung

Zwei Hosts (H1 und H2) mit privaten Adressen werden über Security Gateways (SG1 und SG2) mit öffentlichen Adressen ans Internet verbunden. Die folgende Skizze soll Aufschluss geben, welcher Modus und welche Kombination von IPSec Protokollen innerhalb der VPN-Verbindung verwendet wird.

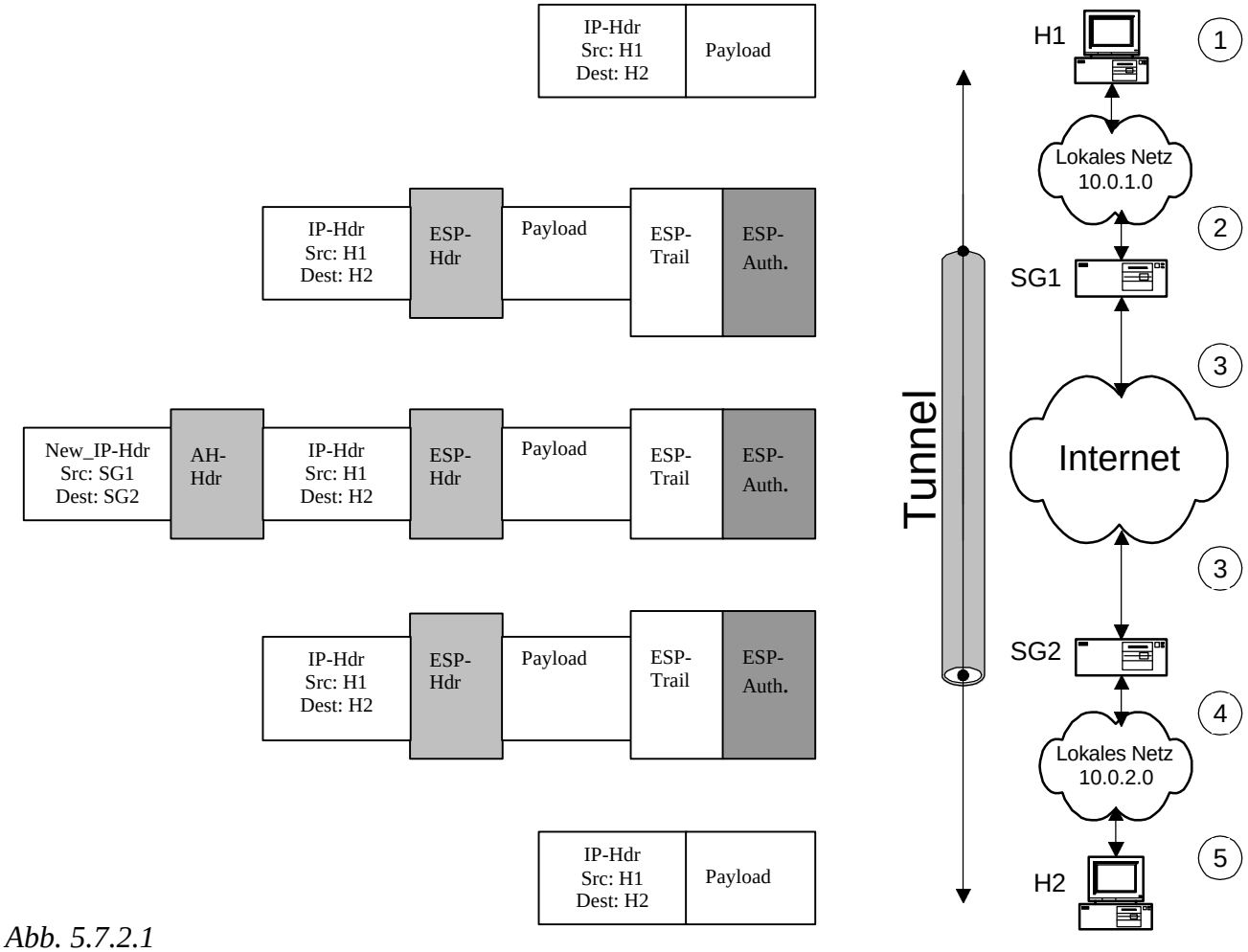


Abb. 5.7.2.1

Beschreibung

H1 erzeugt ein Datenpaket mit Source (Src) und Destination (Dest.) Adresse im IP-Header und dem Payload im Klartext. *Private IP-Adressen* werden für die Adressierung verwendet.

Bevor H1 die Daten auf das lokale Netz (10.0.1.0) sendet, wird das Datenpaket mittels dem ESP-Protokoll bearbeitet. An dieser Stelle könnte man sich fragen, ob es wirklich nötig ist ESP im Transportmodus anzuwenden, den in der Regel betrachtet man die lokalen Netze als sicher.

Für optimalen Schutz über das Internet verwenden die Security Gateways den Tunnelmodus und die beiden IPSec-Protokolle AH und ESP zusammen. Durch das Tunneling bleibt die Identität von Host 1 und 2 im Verborgenen, denn diese befindet sich in den Nutzdaten, welche durch das ESP-Protokoll verschlüsselt worden sind. Das einzige, was man jetzt an diesem Paket erkennen kann, sind die öffentlichen IP-Adressen der Security Gateways SG1 und SG2. Ein Sniffer hat somit weder Kenntnis über die Existenz der Subnetze, noch über die angeschlossenen Hosts. Er sieht nur die beiden Security Gateways. Durch den zusätzlichen AH-Header wird die Authentifikaton auf das ganze Datenpaket ausgeweitet, was bekanntermassen nur mit dem ESP-AH nicht gegeben ist.

Beim SG2 angekommen, wird das Paket zuerst mittels AH-Header authentisiert. Falls die Hashwerte für die Authentifizierung vom Sender und Empfänger übereinstimmen, wird das Datenpaket vom SG2 so bearbeitet, dass es wieder die Form hat wie in 2, also sich im Transportmodus mit ESP befindet und so ins Subnetz (10.0.2.0) durchgeroutet wird.

Am Ziel angekommen, wird mittels ESP-AH wieder eine Authentifikaton durchgeführt. Dies Schützt die Daten vor Angriffen aus dem Subnetz. Falls keine Änderung des Datenpakets vorgenommen worden ist, wendet der H1 das ESP-Protokoll an, um so das Datenpaket zu entschlüsseln. Jetzt liegt das ursprüngliche Datenpaket authentisiert und entschlüsselt für weitere Verarbeitungen für die oberen Layer zur Verfügung.

5.8 IKE Internet Key Exchange

Ein sehr heikles Thema bei jeder Verschlüsselung und Authentifizierung ist die Erzeugung und Geheimhaltung der notwendigen Schlüssel, sprich das Schlüsselmanagement. Die verwendeten Algorithmen können auch noch so sicher sein, wenn die Geheimhaltung der Schlüssel unsicher ist, nützt auch der ausgereifteste Verschlüsselungsalgrorithmus nichts. Genau diese Sicherheit wird durch IKE geboten, wenigstens was die Erzeugung anbelangt.
Wie wir noch sehen werden, kann man auf zwei Arten eine sichere Verbindung übers Internet mit IPSec aufbauen und zwar mit ‘Manueller Schlüsselverbindung’ und ‘Automatischer Schlüsselverbindung’. Da in grösseren VPN’s mit mehreren Benutzern der Administrative Aufwand in Grenzen gehalten werden möchte, wird der ‘Automatischer Schlüsselverbindung’ mittels IKE ganz klar bevorzugt.
Bevor IKE näher betrachtet wird, soll für das bessere Verständnis des Protokolls das Prinzip von Diffie-Hellman erklärt werden, das ein wichtiger Bestandteil davon ist. Eingesetzt wird dieses Verfahren einerseits als Inputparameter bei der Schlüsselerzeugung und anderseits für das Sicherstellen von *Perfect Forward Secrecy.*

5.8.1 Diffie-Hellman

Diffie Hellman gehört zu den *Public-Key-Systemen*, welcher zum Schlüsselaustausch verwendet wird. Mit Diffie Hellman wird das Problem gelöst, das zwei Personen ein Geheimnis, also in diesem Fall einen Schlüssel, vereinbaren können, auch wenn sie es laut und vor aller Augen der Öffentlichkeit tun müssen, sprich übers Internet. Anhand eines Beispiels soll der Algorithmus von Diffie-Hellman erläutert werden.

Ausgangslage:
Alice und Bob wollen einen gemeinsamen geheimen Schlüssel vereinbaren. Einziges Problem ist der Schlüsselaustausch. Es muss über das unsichere Internet geschehen, wo bekannterweise das Sniffen von Datenpaketen im Netz kein Problem ist.

Voraussetzung:
Alice und Bob verfügen je über zwei Schlüssel. Einer wird als privater Schlüssel bezeichnet und der andere als öffentlicher Schlüssel. Der private Schlüssel muss geheim gehalten werden. Der öffentliche Schlüssel kann für jedermann zugänglich sein.

Person	Privater Schlüssel (512 Bit, je grösser umso besser)	Öffentlicher Schlüssel (grosse Primzahl > 10^{100})
Alice	x	n
Bob	y	g

Damit jetzt Alice und Bob einen geheimen Schlüssel über das Internet vereinbaren können, braucht es nur zwei Datentransfers. Innerhalb dieser zwei Datentransfers befinden sich die nötigen Informationen, damit Alice und Bob ihren geheimen Schlüssel erzeugen können. Dieses Prinzip ist so raffiniert, dass ein Spion, obwohl er diesen Datentransfer aufzeichnet, keine Chance hat, den geheimen Schlüssel herauszufinden. Raffiniert oder?

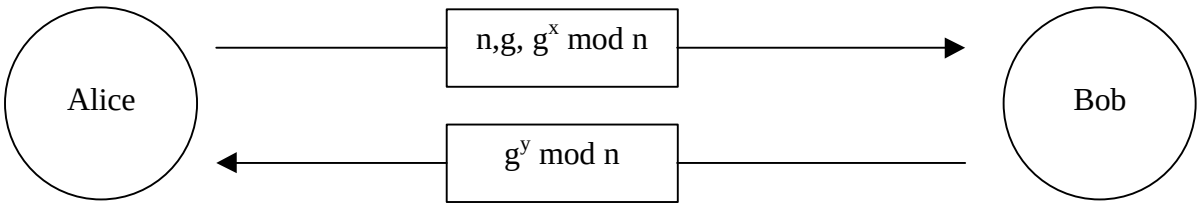


Abb. 5.8.1.1

Wie man sieht, wird im ersten Datentransfer die beiden öffentlichen Schlüssel und das Resultat des Rechenausdrucks ‘g^x mod n’ gesendet. Mit Hilfe dieser Zahl wird dann der geheime Schlüssel berechnet. Bob sendet nun seinerseits den gleichen Rechenausdruck, aber mit dem Unterschied, dass der Exponent diesmal sein privater Schlüssel ist. Nachdem nun beide im Besitz dieser beiden Resultate sind, potenzieren sie es noch mit ihren privaten Schlüsseln. Alice berechnet also $(g^x \bmod n)^y$ und Bob $(g^y \bmod n)^x$. Nun gilt folgender Zusammenhang:
Nach dem Gesetz der modularen Arithmetik ergeben beide Berechnungen ($g^{yx} \bmod n$) . Somit haben Alice und Bob einen gemeinsamen geheimen Schlüssel erzeugt.

Solange kein Algorithmus existiert, welcher aus den Rechenausdrücken ‘g^x mod n’ und ‘ g^y mod n’ die privaten Schlüssel x und y berechnet, kann man den Diffie-Hellman für die Erzeugung eines geheimen Schlüssels verwenden.

5.8.2 Aufgabe von IKE

Nachdem nun das Prinzip von Diffie-Hellman erklärt worden ist, wird jetzt näher auf das IKE Protokoll eingegangen. IKE‘s Hauptaufgabe ist es, das automatische Realisieren der folgenden drei Komponenten für ein sicheres Schlüsselmanagemant:

- Authentifikation des Kommunikationspartners

  IKE bietet vier verschiedene Authenitfikatonsmethoden an, um sicherzustellen, dass man auch wirklich mit der richtigen Person kommuniziert.

  - Authentifikation mit Pre-Shared-Key
  - Authentifikation mit Digitaler Signatur
  - Authentifikation mit Public-Key-Verschlüsselung
  - Verbesserte Methode für Authentifikation mit Public-Key-Verschlüsselung

  In Rahmen dieser Arbeit gehen wir auf die Authentifikationmethoden mit Pre-Shared-Key und Digitaler Signatur näher ein.

- Erzeugen der *Security Association SA*

- Schlüsselerzeugung und Regenerierung
  Während des IKE Vorganges werden mehrere Schlüssel erzeugt [→ Automatische Schlüsselverbindung]. Folgende Input-Parameter werden für das Erzeugen der Schlüssel benützt:
  - Einige Parameter aus den ersten vier Datenpaketen, welche durch die zukünftige sichere Verbindung zwischen den Hosts ausgetauscht werden.
  - Wahl der Authentisierungsmethode ergibt je nach dem einen anderen Input-Parameter
  - Parameter, welche mittels Diffie Hellman erzeugt werden

  Der Verschlüsselungsschlüssel und Authentifizierungsschlüssel für die eigentliche Nutzdatenübertragung werden dann aus diesen Schlüsseln abgeleitet. Das Regenerieren der Schlüssel nach einer bestimmten Zeit erhöht zusätzlich den Sicherheitsgrad.

5.8.3 ISAKMP/Oakley Protokoll

IKE basiert auf zwei Protokollen, nämlich ISAKMP (Internet Security Association and Key Management Protocol) und Oakley (Name des Entwicklers). ISAKMP gibt dem Rahmen vor für die Authentifikation und den Schlüsselaustausch. Es gibt keine Auskunft, wie es implementiert werden soll. Somit können verschieden Verfahren benützt werden, welche die Rahmenvorstellung von ISAKMP erfüllen. ISAKMP wird in zwei Phasen realisiert:

- Phase 1 erzeugt ISAKMP-SA, also diejenige *Security Associtaton*, welche zuständig ist für die Verschlüsselung der ISAKMP-Datenpaket selbst.
- Phase 2 erzeugt die IPSec-SA, die auf die Nutzdaten angewandt wird.

Da ISAKMP den Rahmen vorgibt, übernimmt das Protokoll Oakley die konkrete Ausführung der zwei Phasen. Oakley besitzt folgende drei Modi:

Modus	Anwendung	Aufgabe
Main Modus	Phase 1	Erzeugt ISAKMP-SA
Aggressive Modus	Phase 1	Erzeugt ISAKMP-SA schneller als im Main Modus, aber auf Kosten der Sicherheit
Quick Modus	Phase 2	Erzeugt IPSec-SA

Beim Verbindungsaufbau zwischen zwei Gateways mit IPSec wird die Phase 1 nur einmal durchgespielt. Hingegen die Phase 2 kann mehrere Male für das Regenerieren der Schlüssel, welche in der IPSec-SA angewandt wird, wiederholt werden.

Wie man gesehen hat, authentifiziert IKE Kommunikationspartner, erzeugt und regeneriert Schlüssel und zusätzlich bietet es Schutz gegen verschiedene Attacken, wie

- Denial-of-Service [→ Angriffe im Netz]
- Man-in-the-Middle [→ Angriffe im Netz]

5.9 Linux FreeS/WAN

Der Initiant von Linux FreeS/WAN ist ein Mann, der es sich zum Ziel gesetzt hat, das Recht auf freie Meinungsäusserung sowie auf Vertraulichkeit der Kommunikation über das Internet zu verwirklichen und zu fördern . Sein Name lautet John Gilmore und hat durch seine frühe Tätigkeit bei Sun und anderen Firmen finanzielle derart ausgesorgt, dass er es sich leisten konnte, Zeit und Energie in dieses Projekt zu investieren. Massgeblich Unterstützt wird er von der Linux Gemeinschaft.
Von Anfang an sollten die verwendeten Komponenten, um ein Virtuelles Privates Netz aufzubauen, folgende Eigenschaften aufweisen:

- Lauffähig auf herkömmlichen billigen PC's, keine teuren Hardwarekomponenten müssen angeschafft werden.
- Verwendete Software soll unentgeltlich frei verfügbar sein, also auf dem Prinzip von Linux aufbauen, das als Betriebssystem verwendet wird. Dieser Umstand kommt auch ganz klar im Projektnamen mit 'Linux Free' zum Ausdruck.
- Verwendete Verschlüsselungsalgorithmen und deren Länge der Schlüssel unterstehen keinen Einschränkungen von Export Gesetzen eines Landes. Man muss wissen, dass Verschlüsselungsalgorithmen ab einer bestimmten Länge des Schlüssels unter das Waffengesetz bestimmter Länder fallen. Zum Beispiel in Amerika ist das Exportieren von grösser als 56 Bit Schlüsseln verboten. Zusätzlich ist es US-Bürgern nicht gestatten Verschlüsselungscode zu schreiben, welche den Waffengesetzen zuwiderlaufen weder innerhalb noch ausserhalb der USA, den auch das Gehirn eines USA Bürgers wird als US-Territorium betrachtet.
  Im EU-Raum gibt es das Wassenaar Abkommen, das sagt aus, dass Verschlüsselungsalgorithmen ab einer Länge von 64 Bit Exportkontrollmechanismen unterliegen. Aus diesem Grund sind alle Programmteile in Ländern geschrieben aus denen sie ohne jegliche Restriktionen exportiert werden können.

Die Skizze illustriert den Aufbau von FreeS/WAN

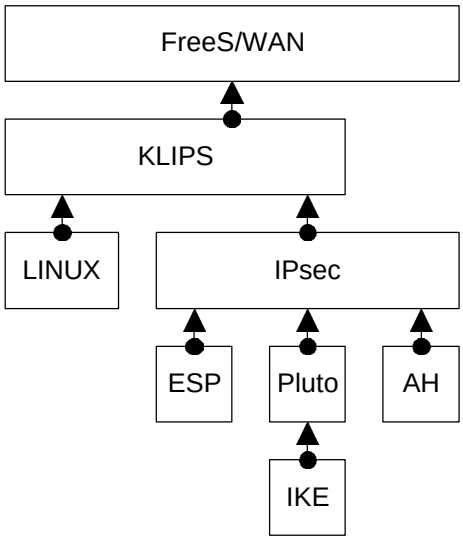


Abb. 5.9.1

5.9.1 FreeS/WAN

Der Hinweis, dass FreeS/WAN auf dem IPSec-Protokoll aufbaut, wird durch den Term im Namen 'S/WAN' ausgedrückt. S/WAN (Secure Wide Area Networks) ist ein anderes Projekt von diversen Herstellern von Firewalls und anderen Sicherheisprodukten, welche sich zusammengeschlossen haben, um aufbauend auf dem Protokoll IPSec, die verschiedenen Implementationen der entsprechenden Hersteller so anzupassen, dass ein Kunde nicht mehr abhängig von einem Produkt ist, um ein Virtuelles Privates Netz aufzubauen. Praktisch bedeutet es, dass die Securtiy Gateways von einem VPN nicht mehr vom selben Hersteller sein müssen. Mit dem Einbinden von S/WAN wird darauf Aufmerksam gemacht, dass FreeS/WAN voll kompatibel zu den Produkten der Hersteller ist, welcher sich am S/WAN beteiligt haben und das es somit zum Aufbau eines VPN geeignet ist.

5.9.2 KLIPS (Kernel IP Security)

KLIPS ist das Werkzeug, welcher im Linux-Kernel die nötigen Veränderung macht, um IPSec in das Linux-Betriebssystem einzubinden, so dass sich Linux FreeS/WAN ergibt, mit dem man dann die Möglichkeit hat ein VPN [→ Installation von FreeS/WAN] zu konfigurieren.

5.9.3 Pluto

Beim IPSec fällt auf, dass anstatt IKE als Schlüsselmanagement-Protokoll ein Protokoll namens Pluto eingebunden wird. Pluto ist nichts anderes als eine 'IKE-light' Version und wird als IKE-Daemon bezeichnet. Pluto bietet also nicht alle Funktionen, welche IKE bietet, aber genau soviel, wie es für das Herstellen eines VPN braucht.

5.10 Was ist in FreeS/WAN implementiert?

In der gegenwärtigen offiziellen Version von Pluto wird in der ersten Phase des ISAKMP-Protokolls der 'Aggressive Mode' nicht unterstützt. Im Entwicklungsteam ist eine Diskussion im Gange, ob es aus Sicherheitsgründen überhaupt jemals in die offizielle Version implementiert werden soll oder nicht. Zurzeit ist der 'Aggressive Mode' nur in Form eines Patches ins System integrierbar. Das Schlüsselmanagement wird somit mit dem Main und Quick Modus ausgeführt.
FreeS/WAN bietet zwei Methoden von Verbindungsaufbau: die manuelle- und die automatische Schlüsselvereinbarung. Bei der manuellen Methode sind die Schlüssel von vornherein bekannt und bei der automatischen werden sie während des Verbindungsaufbaus generiert. Als Verschlüsselungsalgorithmus wird der *3DES-CBC* und als Authentisierung werden die folgenden zwei Algorithmen verwendet:

Authentisierungsalgorithmus	Schlüssellänge in Bit	Hashlänge in Bit
HMAC-SHA-1-96	128	160
HMAC-MD5-96	128	128

6 System-Design

6.1 Netzwerk-Modell

Um ein VPN-Netzwerk mit all seinen Eigenschaften und Anwendungsformen zu realisieren, muss ein geeignetes Modellnetzwerk aufgebaut werden. Uns standen zu Beginn vier Pentium-Rechner (450 Mhz) zur Verfügung. Auf allen Rechner wurde das Betriebssystem Linux installiert, auf zwei Computern zusätzlich Windows NT. Alle PC's wurden mit je zwei Netzwerkkarten ausgestattet. Dies bietet die optimale Grundlage für diverse Experimente.
Für die vier Rechner wurden DNS-Namen (KSY007-KSY010) und IP-Adressen (160.85.131.59-62) aus dem Schulnetz der ZHW reserviert.

Hardware: 4 x Pentium 450 MHz
1 x Pentium 90 MHz
Netzwerk-Analyzer Wandel&Golterman DA-320
10-Base-T-Hub Planet
Analoges Modem 56k Zyxel

Software: Alle Rechner: SuSE Linux 6.2 Kernel-Version 2.2.10
KSY007, KSY010: Linux und Windows NT 4.0

Für das bessere Verständnis wurden für die Rechner symbolische Namen eingeführt.

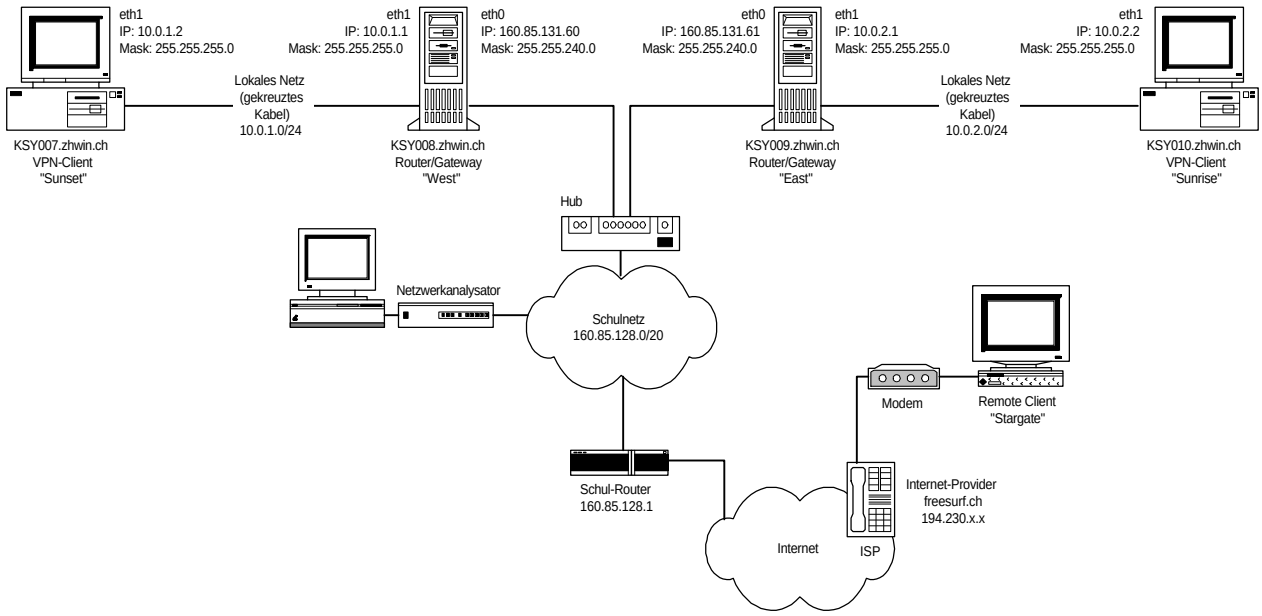


Abb. 6.1

6.2 Vorgehensweise

Nach Installation der Betriebssysteme wurden die Rechner nach Abbildung 6.1 vernetzt. Bevor nun des erste VPN eingerichtet werden kann, muss die volle Funktionalität sichergestellt werden [→ Routing]. Als Basissystem wurde zuerst eine Site-to-Site-Verbindung [→ VPN-Architektur] angestrebt. Das heisst, die beiden Rechner 'West' und 'East' werden als Security-Gateways, und 'Sunrise' und 'Sunset' als VPN-Clients eingerichtet. Zu einem späteren Zeitpunkt wurde dann ein Netzwerk-Analysator installiert, um Datenpakete analysieren zu können. Bald waren Versuche über einen Internet-Provider vorgesehen, was mit einem weiteren PC und analogem Modem realisiert werden konnte. Als ISP wurde der Gratis-Einwahlknoten von Sunrise Communications (freesurf.ch) benutzt.

6.3 Routing

Voraussetzung für das Installieren von FreeS/WAN ist, die Verbindungen zwischen den Stationen funktionieren, So können spätere Fehlerquellen besser lokalisiert werden. Damit dies möglich wird, muss man folgende Arbeiten ausführen:

- West und East als Router konfigurieren
- Statische Routing-Tabellen von West und East entsprechend ergänzen
- Testen der Verbindungen mit Ping-Befehl

6.3.1 Clients

Die Routing-Tabellen sind unter Linux in der Datei *etc/route.conf* zu finden. Obwohl die VPN-Clients nicht als Router sondern als Hosts konfiguriert sind, werden beim Einrichten der Netzwerkkarten Routing-Tabellen generiert. Damit wird der Datenfluss einem bestimmten Netzwerk-Device zugeordnet. Hier müssen keine Veränderungen an den Routing-Tabelle vorgenommen werden. Der Vollständigkeit halber, sind die entsprechenden Routing-Tabellen von Sunset und Sunrise hier aufgeführt.

 # 'Sunset'

```
Destination       Gateway          Netmask          Device
10.0.1.0          0.0.0.0          255.255.255.0    eth1
default           10.0.1.1
```

 # 'Sunrise'

```
Destination       Gateway          Netmask          Device
10.0.2.0          0.0.0.0          255.255.255.0    eth1
default           10.0.2.1
```

6.3.2 IP-Forwarding

Anders ist es bei den Security-Gateways. Diese müssen als Router konfiguriert werden, indem IP-Forwarding aktiviert wird. Diese Einstellung wird in der Datei *etc/rc.config* vorgenommen:

```
# runtime-configurable parameter: forward IP packets.
# Is this host a router? (yes/no)
#
IP_FORWARD="yes"
```

6.3.3 Gateways

Da wir in unserem Modell statische Routing-Tabellen verwenden, müssen diese manuell ergänzt werden. Dazu editiert man die Datei *etc/route.conf* in einem Editor und fügt die gewünschten Einträge ein. Das ist notwendig, damit die zwei Subnetze Datenpakete miteinander austauschen können. Die Routing-Tabellen der Gateways West und East sehen dann folgendermassen aus:

 # 'West'

Destination	Gateway	Netmask	Device
160.85.128.0	0.0.0.0	255.255.240.0	eth0
10.0.1.0	0.0.0.0	255.255.255.0	eth1
10.0.2.0	**160.85.131.61**	**255.255.255.0**	**eth0**
default	160.85.128.1		

 # 'East'

Destination	Gateway	Netmask	Device
160.85.128.0	0.0.0.0	255.255.240.0	eth0
10.0.2.0	0.0.0.0	255.255.255.0	eth1
10.0.1.0	**160.85.131.60**	**255.255.255.0**	**eth0**
default	160.85.128.1		

Diese Einträge gewährleisten nun den Pakettransfer zwischen den beiden lokalen Netzen. Nehmen wir an, ein Paket wird von Sunset wird nach Sunrise geschickt [→ Abbildung 6.1]. Die Zieladresse ist somit 10.0.2.2. Da sich diese Zieladresse nicht im eigenen Subnetz befindet, wird das Paket über den Standard-Gateway geschickt, in diesem Fall ist das der Rechner West. Dort wird die Zieladresse mit der Netmaske logisch AND-verknüpft, um die Netzadresse des Ziel-Netzwerkes herauszufinden. Nach dieser Prozedur wird nach der Netzadresse in der Destination Tabelle gesucht. Falls der Router für diese Netzadresse keinen Eintrag hat, schickt er das Paket über seinen Standart-Gateway, also über eth0 an die Adresse 160.85.128.1. In unserem Fall findet er nun das Netz 10.0.2.0 und sendet es auf der Netzwerkkarte eth0 zum nächsten Router mit der Adresse 160.85.131.61. Dieser erennt, dass das Paket zu seinem Subnetz gehört und findet leitet es anhand der Host-Adresse, in unserem Fall die 2, an den richtigen Client. Der praktische Test für diese Verbindung wird nun im weiteren abgehandelt.

6.3.4 Testen der Verbindung

Die Verbindung zwischen den beiden Clients 'Sunrise' und 'Sunset' kann jetzt mit dem Ping-Befehl getestet werden. Zur Ausführung des Befehls wird folgende Terminaleingabe gemacht:

```
ping 10.0.2.2
```

Sunrise richtet nun eine Anfrage (Echo-Request) an Sunset und erwartet eine Antwort (Echo-Replay) von ihm. Falls eine Antwort zurückkommt, funktioniert die Verbindung zwischen den Clients. Ansonsten müsste man die Routing-Tabellen, Kabel, Stecker usw. kontrollieren.

7 FreeS/WAN

7.1 Der Linux-Kernel

Der Unix-Kernel stellt eine Art Vermittler zwischen den Anwenderprogrammen und der Hardware des Computers dar. Er verwaltet den Arbeitsspeicher des Rechners und sorgt dafür, dass jedes laufende Programm angemessene Anteile der Prozessor-Arbeitszyklen zugewiesen bekommt. Insgesamt stellt der Kernel dabei eine portable Schnittstelle zur eigentlichen Hardware dar.

7.1.1 Erzeugung eines eigenen Kernels

Nach der Installation von Linux befindet sich der Kernel im `root`-Verzeichnis (`vmlinuz`). Da der Kernel nicht immer optimal auf das System abgestimmt ist, empfiehlt es sich in einigen Fällen einen eigenen Kernel zu generieren. Für diesen Zweck stehen make-Befehle zur Verfügung. Somit wird der Quellencode mit dem C-Compiler übersetzt. Die Kernelquellen befinden sich im Verzeichnis `/usr/src/linux`. Vor der Kompilierung wird der Kernel konfiguriert. Dazu stehen drei verschiedene Methoden zur Verfügung.

1. `make config` Konfiguration über Kommandozeile
2. `make menuconfig` Durch Menü im Textmodus
3. **`make xconfig`** Durch Menü unter dem X Window System

Da wir auf unserem System mit graphischer Oberfläche (KDE) arbeiten, benützen wir die komfortable Variante `xconfig`. Das Konfigurationsscript enthält zahlreiche Fragen, die im Normalfall mit y (Ja) oder n (Nein) beantwortet werden können. Viele Treiber bieten ausserdem die Option m an, dies steht für Modul. Dadurch wird der Treiber zwar kompiliert, aber nicht direkt in den endgültigen Kernel eingebunden, sondern als ladbares Modul bereitgestellt.

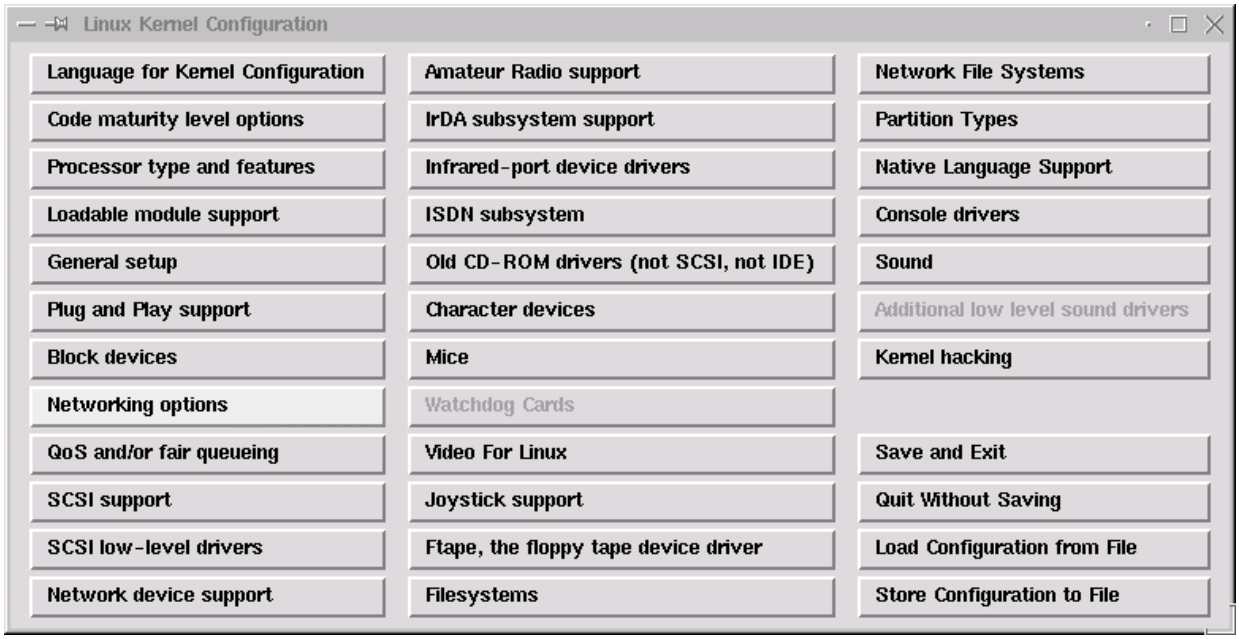


Abb. 7.1 Konfiguration des Kernels im X Window System

Nach der Konfiguration wird die eigentliche Kompilierung gestartet. Dies geschieht durch folgenden Befehls-String:

```
/usr/src/linux # make dep clean bzImage
```

Dabei werden die wechselseitigen Abhängigkeiten der Quell- und Include-Dateien richtig zusammengestellt (dep), alle alten Objekt-Dateien gelöscht (clean), und der neue, komprimierte Kernel in die Datei bzImage im Verzeichnis `arch/i386/boot` geschrieben. Je nach Leistung des Systems dauert dieser Vorgang zwischen 4 Minuten (Pentium) und einigen Stunden (386er). Anstelle von zImage haben wir bzImage benützt, weil der Kernel auf Grund der vielen Features zu gross geworden ist.

Die Installation des Kernels erfolgt durch das Kopieren der Datei bzImage nach vmlinuz in der boot-Partition. Es ist sehr wichtig, dass zuvor eine Sicherheitskopie des alten Kernels, sowie eine Bootdiskette erstellt wird, denn irgendwann kommt immer der Tag, an dem man aus Versehen den Kernel von der Festplatte löscht oder eine ähnliche Dummheit begeht.
Zum Schluss muss LILO (Linux Loader) neu installiert werden, damit der neue Kernel auch gebootet wird:

```
/usr/src/linux # make bzlilo
```

Nun kann der Rechner neu gestartet werden.

7.2 Installation von FreeS/Wan

Da die IPSec-Funktionen auf dem System wirksam zu machen, muss das Programm in den Kernel eingebunden werden. Dazu sind die folgenden Schritte nötig:
Die komprimierte Datei *snapshot.tar* wird ins Verzeichnis */usr/src* kopiert. Von dort aus wird sie mit all ihren Unterverzeichnissen entpackt:

```
tar xvfz snapshot.tar
```

Aufruf des Konfigurationsprogramms im X Windows System:

```
/usr/src/FreeSWan-Verzeichnis # make xgo
```

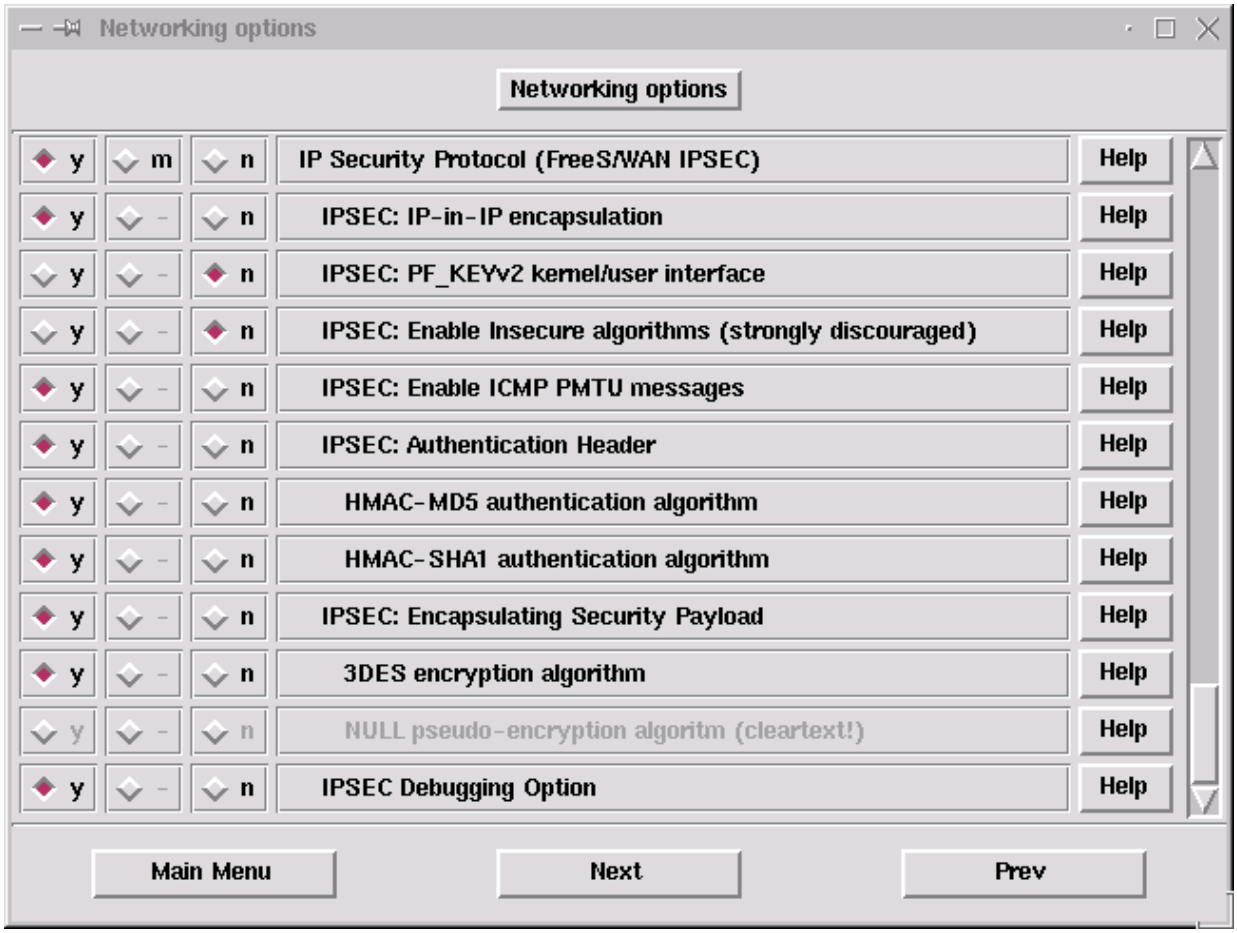


Abb. 7.2 IPSec-Optionen im Konfigurationsmenu

In der Gruppe *Networking Options* muss sichergestellt werden, dass die folgenden Menüpunkte aktiviert sind:

- IP: tunneling
- Kernel/User network link driver
- IP: optimize as router not host
- IP Security Protocol (FreeS/Wan IPSEC)

Zudem muss sichergestellt sein, dass IP-Forwarding eingeschaltet ist. Dieser Eintrag kann in der Datei */usr/etc/rc.config* gemacht werden (`IP_FORWARD="yes"`).

Wenn man die Kernelkonfiguration mit *Save and Exit* verlässt, wird die Kompilierung automatisch gestartet. Dieser Vorgang dauert einige Minuten. Danach muss der neue Kernel noch ins `boot`-Verzeichnis kopiert werden (→ Linux-Kernel). Entweder manuell oder mit dem Befehl:

```
/usr/src/linux # make install
```

Zum Schluss muss LILO (Linux Loader) neu installiert werden, damit der neue Kernel auch gebootet wird:

```
/usr/src/linux # make bzlilo
```

Nun kann der Rechner neu gestartet werden.

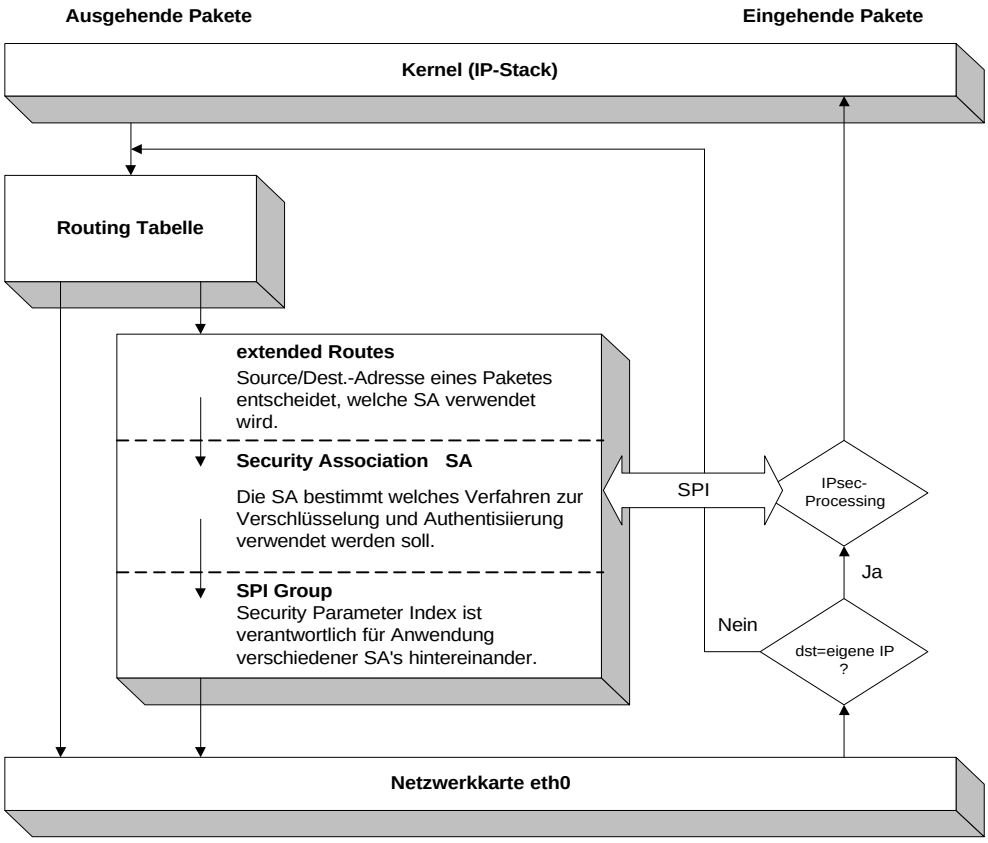


Abb. 7.3 Die Grafik zeigt wie IPsec zwischen dem Kernel und der Netzwerkkarte arbeitet.

7.3 Konfiguration von FreeS/WAN

Nachdem das Netzwerk aufgebaut, alle Verbindungen getestet und FreeS/WAN installiert worden ist, kann mit der Konfiguration von FreeS/WAN begonnen werden. Die Konfiguration spielt sich hauptsächlich auf den folgenden zwei Dateien ab:

```
/etc/ipsec.conf
/etc/ipsec.secrets
```

7.3.1 Ipsec.conf

Die Datei `ipsec.conf` wird verwendet, um die Verbindungsvereinbarungen zwischen zwei Stationen festzulegen. Diese Datei kann auch als *Security Association* bezeichnet werden. Diese Datei beinhaltet zwei Teile. Im ersten Teil legt sie den ‘config setup’ fest und im zweiten Teil die ‘connection specifications’. Der vollständige Inhalt dieser Datei ist im Anhang [xy] einsehbar. Die nötigen Einträge, welche dort fett gekennzeichnet sind, werden hier im einzelnen wiedergegeben. Alle anderen gesetzten Parameter sind standardmässig eingestellt. Mehr Information zu den einzelnen Einstellungen kann man in der ‘man- Page’ von ipsec.conf nachlesen.

```
# /etc/ipsec.conf - FreeS/WAN IPSEC configuration file

config setup                          (1.Teil)
      interfaces="ipsec0=eth0"

# connection specifications           (2. Teil)
conn ‘West-East‘
      # left security gateway (public-network address)
      left=160.85.131.60
      # subnet behind left (omit if there is no subnet)
      leftsubnet=10.0.1.0/24
      # right security gateway (public-network address)
      right=160.85.131.61
      # subnet behind right (omit if there is no subnet)
      rightsubnet=10.0.2.0/24
```

Im ‘config setup’ gibt es vorerst nur eine Zeile, welche verändert werden muss, und zwar ist es die Angabe der Netzwerkkarte, welche dem IPSec-Device zugeortnet wird.
Im Abschnitt ‘connection specifications’ gibt es einige Änderungen, welche man vornehmen muss. Im ersten Schritt muss man bei der Zeile ‘conn’ einen Verbindungsnamen angeben. In unserem Fall ist er durch die Namen der Security-Gateways ‘West-East‘ gegeben. Als nächstes werden die vier IP-Adressen von den involvierten Stationen angegeben. Bezogen auf unser Modell wird ‘left‘ für ‘Sunset‘ und ‘West‘ interpretiert und ‘right‘ für ‘East‘ und ‘Sunrise‘. Die anderen standardmässig eingetragenen Optionen können vorerst unverändert gelassen werden.

7.3.2 Ipsec.secrets

Dieses File beinhaltet die Angaben für eine richtige Authentifikation zwischen zwei Stationen. Dazu gehören die IP-Adressen der Kommunikationspartner sowie ihr gemeinsames Geheimnis in Form eines 256 Bit Schlüssels. Es können durchaus mehrere Geheimnisse mit verschiedenen Partnern abgemacht werden. In unserem Fall sind es ‘West’ und ‘East‘. Wie man weiter unten sieht, werden zuerst die IP-Adressen angegeben und auf der nächsten Linie steht das 256 Bit lange gemeinsame Geheimnis. Falls also dieser Schlüssel auf der Verbindung ‘West-East‘ nicht auf beiden Stationen das gleiche ist, wird keine Verbindung aufgebaut. Standardmässig wird bei der Installation von FreeS/WAN ein zufälliger 256 Bit Schlüssel erzeugt, so dass man nur noch die IP-Adressen anpassen muss.

```
# /etc/ipsec.secrets - FreeS/WAN IPSEC authentification file

160.85.131.60 160.85.131.61

"0xba078756_76cb52c6_1ca1bd29_2b775e0d_7cd2ac38_8718463f_3542049d_da1a7a72"
```

7.4 Verbindungsaufbau mit IPSec

Nach diesen Modifikationen dieser zwei Dateien ist es nun möglich eine sichere Datenverbindung aufzubauen. Es gibt nun zwei Arten eine sichere Verbindung aufzubauen.

- Manuelle Schlüsselverbindung
- Automatische Schlüsselverbindung

7.4.1 Manuelle Schlüsselverbindung

Voraussetzung für eine manuelle Schlüsselverbindung ist die vorherige Kenntnis aller nötigen Informationen, so dass keine vorherige Verbindungsaufbau-Abmachung mehr zwischen den beteiligten Security Gateways getroffen werden müssen. Alle nötigen Informationen befinden sich in der `/etc/ipsec.conf` Datei.

Hier ist der Abschnitt mit den Angaben, welche in der Datei `ipsec.conf` im Abschnitt "connection specifications" für 'manual' definiert sind.

```
# (manual) encryption/authentication algorithm and parameters to it
esp=3des-md5-96
espenckey=0xf83f6c87_f84b9ae4_43764981_86544c6f_c1d4d360_af0b860e
espauthkey=0xd37984f1_13070cf2_e0eaf4f8_01cd414b
```

esp	Daraus wird ersichtlich, dass das IPSec-Protocol ESP [→ 5.6 ESP] , als Verschlüsselungsalgorithmus *3DES-CBC* und als Authentisierungsverfahren *HAMAC-MD5-96* verwendet wird.
Espenckey	ESP encoding key, mit diesem 192 Bit langen Schlüssel S_{ESP} in Hex-Code werden die *3DES-CBC* Daten verschlüsselt.
Espauthkey	ESP authentication key, mit diesem 128-Bit langen Schlüssel S_{AUTH} wird der 96 Bit lange *Hashwert* berechnet.

Um eine manuelle Verbindung aufzubauen muss der folgende Befehl eingegeben werden:

<ipsec manual –show --up name> , also in unserem Fall:

```
ipsec manual --show --up West-East
```

Wenn man Ipsec manual abschalten möchte, muss man folgenden Befehl eingeben:

<ipsec manual --down name> , also in unserem Fall:

```
ipsec manual --show --down West-East
```

7.4.2 Automatische Schlüsselverbindung

Der Unterschied zur ‘Manuellen Schlüsselverbindung’ ist nur die Schlüsselverteilung. Wie wir gesehen haben ist bei der ‘Manuellen Schlüsselverbindung’ die nötigen Schlüsselangaben in der Datei `ipsec.conf` schon vorhanden. Bei der ‘Automatischen Schlüsselverbindung’ werden die nötigen Informationen für eine sichere Verbindung automatisch im Hintergrund generiert. Wobei die Dateien `ipsec.conf` die nötigen Angaben für einen ‘auto-Start‘ bereithält. In diesem Fall muss man nichts Eingeben, alles ist schon standardmässig eingegeben.

```
# /etc/ipsec.conf - FreeS/WAN IPSEC configuration file

# (auto) key-exchange type
keyexchange=ike
# (auto) key lifetime (before automatic rekeying)
keylife=8h
```

Wie man sieht, geschieht das Schlüsselmanagement mit dem Protokoll ‘ike’[→ 5.8 IKE]. Folgende Aufgaben werden mittels IKE erfüllt:

- Authentifizierung des Gesprächspartners
- Automatisches Erzeugen von Sicherheitsabmachungen (*Security Association SA*)
- Erzeugen der benötigten Schlüssel anhand der SA. Dies geschieht einerseits mittels Diffie-Hellman und anderseits mittels zusätzlichen spezifischen Informationen aus dem Verbindungsaufbau.
- Aus sicherheitstechnischen Gründen ist es von Vorteil, wenn nach einer gewissen Zeit eine neue SA mit dementsprechend neuen Schlüsseln generiert werden kann. IKE stellt auch diese Möglichkeit zur Verfügung. Im obigen Beispiel sieht man, dass ein neuer Chiffrierschlüssel nach 8 Stunden erzeugt wird. Je kürzer die eingetragen Zeit hier ist umso grösser ist der Grad der Sicherheit für die Nutzdaten, aber das geht dann auf Kosten der Übertragungszeit.

Weitere Optionen für die Schlüsselerzeugung und deren Lebensdauer im Auto-Betrieb erhält man anhand des ‘man-Pages‘ mit **`man ipsec_auto`**.

Um eine automatische Verbindung aufzubauen, muss der folgende Befehl bei beiden Security-Gateways ‘West’ und ‘East‘ eingegeben werden:

<ipsec auto –show --add name> , also in unserem Fall:
`ipsec auto --show  --add West-East`

Danach muss **nur auf einer Station**, also auf ‘West’ oder ‘East‘, der folgende Befehl eingegeben werden:

<ipsec auto –show --up name> , also in unserem Fall:
`ipsec auto --show --up West-East`

Falls man Ipsec auto abschalten möchte, muss man folgenden Befehl eingeben:
<ipsec auto --down name> , also in unserem Fall:
`ipsec auto --show  --down West-East`

7.5 Verbindungsaufbau mit IKE

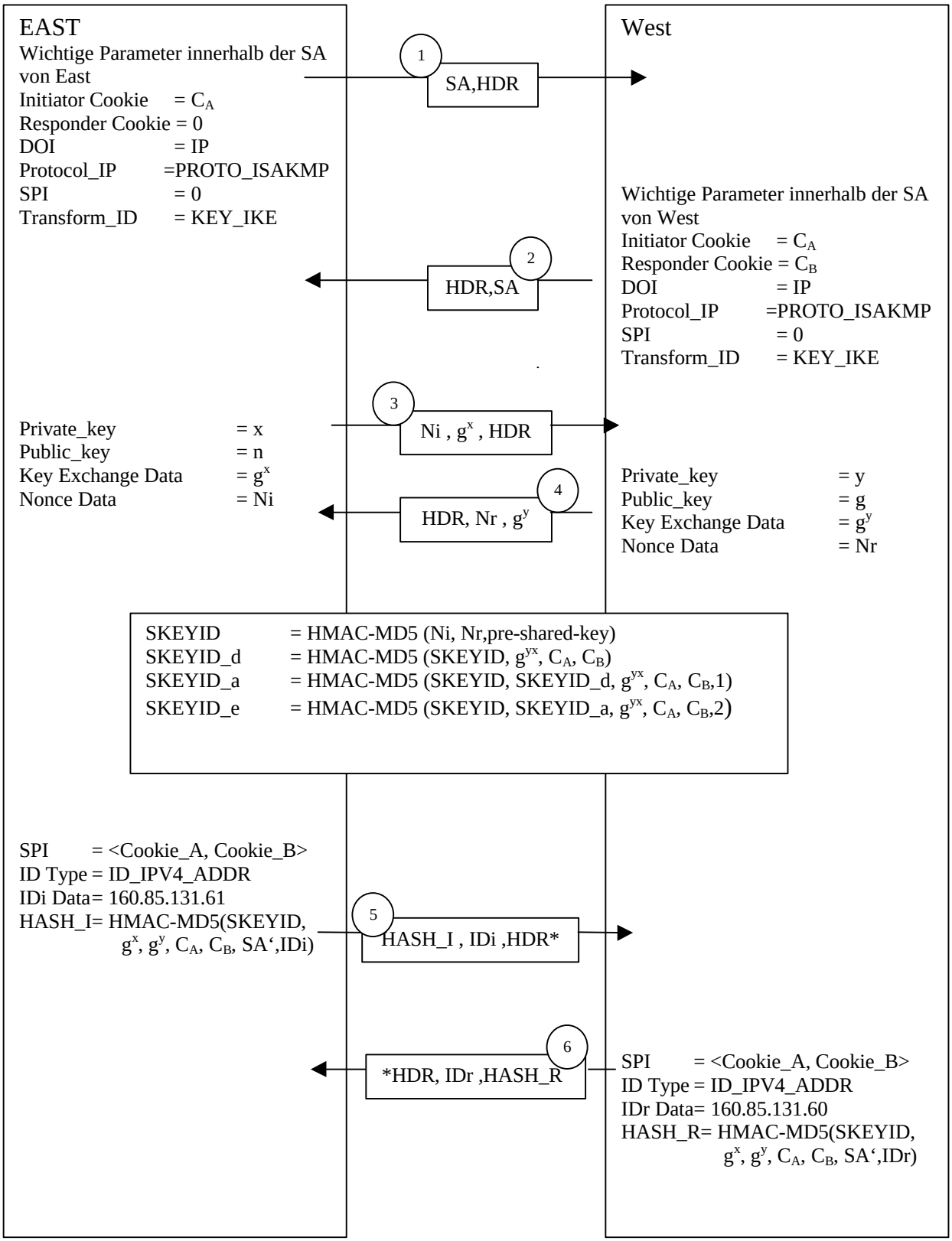
Nach der Eingabe des Befehls <ipsec auto --show --up West-East> werden genau 9 Datenpakete zwischen ‘West’ und ‘East‘ ausgetauscht. Dieser Datenverkehr basiert auf dem IKE-Protokoll. Wie man im IKE Kapitel erfahren konnte, wird damit die zwei Phasen realisiert. Die ersten sechs Datenpakete gehören zum Main Modus und die letzen 3 zum Quick Modus.
Im Main Modus wird festgelegt, mit welchen Verschlüsselungs- und Hashalgorithmen die Datenpakete 5 bis 9 bearbeitet werden. Man sagt, es wird eine ISAKMP-SA ausgehandelt. Des weiteren wird festgelegt, welche Authentifizierungsmethode angewendet wird. Diese Abmachung gilt also nur für den Verbindungsaufbau. Keiner dieser Abmachungen des Main Modus wird für die Nutzdaten angewandt. Zum Schluss der Phase 1 wird die vorher abgemachte Authentifizierung durchgeführt, so dass man sicher sein kann, dass man mit der richtigen Person kommuniziert.
In der zweiten Phase, dem Quick Modus, wird wiederum eine Authentifizierungsmethode, Verschlüsselungs- und Hashalgorithmus festgelegt, aber diesmal sind es Abmachungen, welche die Nutzdaten betreffen. Des weiteren wird der Quick Modus für das Erneuern der Schlüssel für die Verschlüsselung der Nutzdaten verwendet.

Im folgenden sind die entsprechenden ausgetauschten Datenpakete dargestellt, wenn mit einem Sniffer der Verbindungsaufbau zwischen ‘West’ und ‘East’ aufgenommen wird. Den Verbindungsaufbau startet ‘East‘ mit der IP-Adresse 160.85.131.61 zum ‘West’ mit der IP-Adresse 160.85.131.60 .

```
Number  DeltaTime      Source         Destination           Interpretation
# Main Mode, die erste Phase
1       0 ms           160.85.131.61  160.85.131.60         UDP D=500 S=500 Len=184
2       860 us         160.85.131.60  160.85.131.61         UDP D=500 S=500 Len=88
3       26.7 ms        160.85.131.61  160.85.131.60         UDP D=500 S=500 Len=188
4       56.5 ms        160.85.131.60  160.85.131.61         UDP D=500 S=500 Len=188
5       32.5 ms        160.85.131.61  160.85.131.60         UDP D=500 S=500 Len=76
6       420 us         160.85.131.60  160.85.131.61         UDP D=500 S=500 Len=76
# Quick Mode, die zweite Phase
7       29.2 ms        160.85.131.61  160.85.131.60         UDP D=500 S=500 Len=324
8       57.9 ms        160.85.131.60  160.85.131.61         UDP D=500 S=500 Len=300
9       35.6 ms        160.85.131.61  160.85.131.60         UDP D=500 S=500 Len=6
```

Um eine genauere Beschreibung dieses Prozesses wiederzugeben, muss man vorher wissen, welchen Authentifiziermodus das Protokoll IKE verwendet. In dem Softwarpacket FreeS/WAN 1.00, das wir verwenden, gibt es standardmässig im Pluto-Daemon [→ 5.9 Linux FreeS/WAN] nur einen, und zwar die Authentifizierung mit einem “Pre-Shared-Key”. Im späteren Stadium unserer Arbeit werden wir mittels eines OpenSSL–Patchs die Möglichkeit haben, eine Authentifikation mit Digitalen Signaturen durchzuführen.
Jetzt folgt eine detailliertere Beschreibung des Ablaufs von Main und Quick Mode.

7.6 Main Mode (Phase 1)



HDR ISAKMP-Header, dessen nachfolgende Daten unverschlüsselt übertragen werden.

*HDR ISAKMP-Header, dessen nachfolgende Daten verschlüsselt übertragen werden.

Cookie Zufallswert, der einerseits als Schutz gegen ‘Denial-of-Service‘ bieten soll. Mit diesen Cookies von ‘East’ und ‘West’ lässt sich auf schnelle und ohne Prozessor intensive Operationen entscheiden, ob ein Datenpaket ungültig ist und verworfen werden soll oder eben nicht. So lässt sich ein Denial-of-Service Attakke verhindern. Anderseits wird es auch für die Schlüsselerzeugung als Input-Parameter verwendet.

SA Security Association wird als Input Argument für den *Hashwert* mittels *HMAC-MD5* verwendet, welcher wiederum als Input für die Bildung der Signatur benützt wird.

DOI Domain of Interpretation gibt Auskunft über das Format und Aufbau, der im ISAKMP verwendeten Datenpakete, damit sie richtig interpretiert werden können. Ein ISAKMP-Datenpaket kann je nach dem aus verschiedenen Datenpaketen aufgebaut sein. Darin befinden sich dann die Parameter, welche für das Bilden einer *SA* nötig sind. Folgende Datenpakete können innerhalb eines ISAKMP-Datenpakets vorkommen:

- ISAKMP-Header-payload
- Security Association Payload
- Proposal Payload
- Transform Payload
- Key Eychange Payload
- Identification Payload
- Certificate Payload
- Hash Payload
- Signature Payload
- Nonce Payload

Protocol_IP PROTO_ISAKMP, diese Protokoll spezifiziert den Datenschutz während der 1.Phase

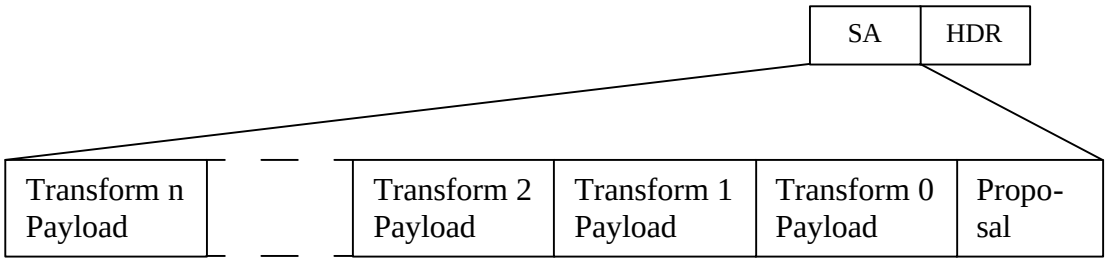
SPI Security Parameter Index, dieser Index kann mit einem Pointer verglichen werden, welcher auf eine SA zeigt. Damit löst man das Problem, mehrere SA’s zwischen zwei Gateways zu unterscheiden. Wenn also ein Datenpaket am Gateway angekommen ist, wird mittels des SPI die richtigen SA gefunden und kann somit mit den richtigen Verfahren das Paket bearbeiten.

Transform_ID Key-IKE, zurzeit ist nur IKE implementiert.

7.6.1 Prozessbeschreibung des Main Mode

Wir gehen nur auf den Teil der Informationen in den Datenpaketen ein, das für ein Verständnis für den Ablauf des IKE Vorganges nötig ist. Mehr Hintergrundinformationen sind den RFC's 2407, 2408 und 2409 zu entnehmen.

Datenpaket 1: Im ISAKMP-Header befindet sich das Cookie C_A. Der SA-Payload von 'East' beinhaltet Vorschläge mit verschiedenen SA's. Das heisst, dass innerhalb des SA-Payloads ein Proposal-Payload existiert mit mehreren Transform-Payloads. Im SA-Payload zwischen 'West-East' waren es 8. Jedes vorgeschlagene Payload besitzt die weiter unten angegebenen Parameter-Angebote. Somit weiss 'West', welche Sicherheits-Algorithmen 'East' unterstützt und kann es somit mit seinen vergleichen und ein Transform-Paket auswählen.



Parameter	Wert	Mögliche Optionen
transform	1	KEY_IKE = 1
OAKLEY_ENCRYPTION-ALGORITHM	5	DES-CBC = 1 IDEA-CBC = 2 BLOWFISH-CBC = 3 RC5-R16-B64-CBC = 4 3DES-CBC = 5 CAST-CBC = 6
OAKLEY_HASH_ ALGORITHM	2	MD5 = 1 SHA = 2
OAKLEY_AUTHENTICATION_ METHODE	3	PRE-SHARED-KEY = 1 DSS-SIGNATURES = 2 RSA-SIGNATURES = 3 ENCRYPTION-RSA = 4
OAKLEY_GROUP_ DESCRIPTION	2	MODP_768 = 1 MODP_1024 = 2 EC2N_GP[2^155] = 3 EC2N_GP[2^185] = 4

Datenpaket 2: ‘West’ sendet einerseits das Transform-Packet zu ‘East’ zurück, für welches es sich entschieden hat. In unserem Fall war es das Transform-Packet 1 mit den oben angegebenen Werten. Im ISAKMP-Header befindet sich jetzt auch der
Eintrag für das Cookie von ‘West’ C_B.

Datenpaket 3/4: Mit diesen Datenpaketen wird mittels Diffie-Hellman [→ 5.8 IKE] ein geheimer Schlüssel generiert. Dieser erzeugte Schlüssel ‘g^{yx} mod n‘ wird als Inputparameter für die Schlüsselerzeugung gebraucht. Des weiteren werden die Zufallswerte Ni und Nr ausgetauscht, die auch für die Schlüsselerzeugung benützt werden.

Je nachdem für welchen Group-Mode sich die Kommunikationspartner einigen, sind die Parameter n und g vordefiniert.

Nach diesen vier Schritten hat das Protokoll IKE die nötigen Informationen um die vier Schlüssel zu generieren, welche auf die ISAKMP Datenpakete und auf die Nutzdaten angewendet werden. Alle Schlüssel werden mit dem *HMAC-MD5* Verfahren unter Verwendung der ausgetauschten und generierten Parameter in den Datenpaketen 1 bis 4 berechnet.

Schlüssel	Zweck
SKEYID = HMAC-MD5 (Ni, Nr,pre-shared-key)	Die Definition dieses Schlüssels ist abhängig von der Authentifikatonsmethode die man gewählt hat. Pre-shared-key: SKEYID = HMAC-MD5 (Ni, Nr,**pre-shared-key**) DSS_signature: SKEYID = HMAC-MD5 (Ni, Nr,**g^{yx} mod n**) usw. Dieser Schlüssel wird wiederum als Input für die nächsten drei Schlüssel verwendet und somit indirekt zum Verschlüsseln irgendwelcher Daten gebraucht.
SKEYID_d = HMAC-MD5 (**SKEYID**, g^{yx}, C_A, C_B)	Dieser Schlüssel dient einerseits wiederum als Input für die Erzeugung des nächsten Schlüssels und anderseits zur späteren Erzeugung von Schlüsseln für die Nutzdaten
SKEYID_a = HMAC-MD5 (**SKEYID**, **SKEYID_d**, g^{yx}, C_A, C_B,1)	Dieser Schlüssel dient einerseits wiederum als Input für die Erzeugung des nächsten Schlüssels und anderseits als Authenfigikatons-Schlüssel für die ISAKMP-Messages
SKEYID_e = HMAC-MD5 (**SKEYID**, **SKEYID_a**, g^{yx}, C_A, C_B,2)	Mit diesem Schlüssel werden die ISAKMP-Datenpakete verschlüsselt.

Ab jetzt werden die Datenpakete (5-9) mit den obigen Schlüsseln authentifiziert und verschlüsselt.

Datenpaket 5/6: Die Authentifizierung passiert mittels *Hashwerten*. Jetzt werden der HASH_I von ‘East’ und der HASH_R von ‘West’ gebildet und den entsprechenden Partnern geschickt und verglichen. Das heisst im Fall von ‘East’, dass es seinen $HASH\_I_{EAST}$ zu ‘West’ mit dem ID-Payload, deren Inhalt seine Identität zeigen soll, zuschickt. In unserem Fall werden die IP-Adressen als Identitätsmerkmal verwendet. Der ‘West’ berechnet auf gleiche Weise wiederum von sich aus den $HASH\_I_{WEST}$ Wert und vergleicht sie. Falls beide gleich sind, wird die ISAKMP-SA als gültig abgeschlossen. Falls sie nicht gleich sind, wird der Verbindungsaufbau abgebrochen. Solange ein Betrüger nicht im Besitze des ‘Pre-Shared-Key‘ ist, bleibt dieser Prozess sicher gegenüber Man-in-the-Middle-Angriffen.
Zum ID-Payload muss noch gesagt sein, dass es verschiedene Arten gibt, wie sich die Hosts identifizieren. Folgende ID-Typen existieren:

ID_IPV4_ADDR:	Verwenden der IP-Adresse. In unserem Fall wird dieser Typ verwendet, also 160.85.131.60
ID_FQDN:	Verwenden von DNS Namen, als “ksy008.zhwin.ch“
ID_USER_FQDN:	Verwenden von Email Adressen wie e5student@zhwin.ch
ID_IPV4_ADDR_SUBNET:	Verwenden der IP-Adresse und deren Subnetzmaske, zum Beispiel IP-Adresse: 10.0.2.1 Subnetzmaske: 255.255.255.0 usw. (Mehr davon RFC 2407/ Seite 19)

7.7 Quick Mode (Phase 2)

Die Phase 2 erzeugt die Ipsec-SA. Es wird als Quick bezeichnet, weil die nötigen Sicherheitsvorkehrungen schon in der Phase 1 abgeschlossen und auf die Phase 2 angewendet werden können. Jetzt folgt eine detaillierte Beschreibung dieses Prozesses.

Bekanntlich wird im Quick Modus 3 Datenpakete ausgetauscht. Für jedes dieser drei wird ein Hashwert für die Authentifizierung erzeugt. Berechnet werden sie folgendermassen:

HASH_1 = HMAC-MD5(SKEYID_a, M_ID, $SA_{Initiator}$, Ni, $KE_{Initiator}$)

HASH_2 = HMAC-MD5(SKEYID_a,Ni, M_ID, $SA_{Responder}$, Nr, $KE_{Responder}$)

HASH_3 = HMAC-MD5(SKEYID_a,0, M_ID, Ni, Nr)

S_KEYID_a: Schlüssel, welcher in der Phase 1 erzeugt worden ist.

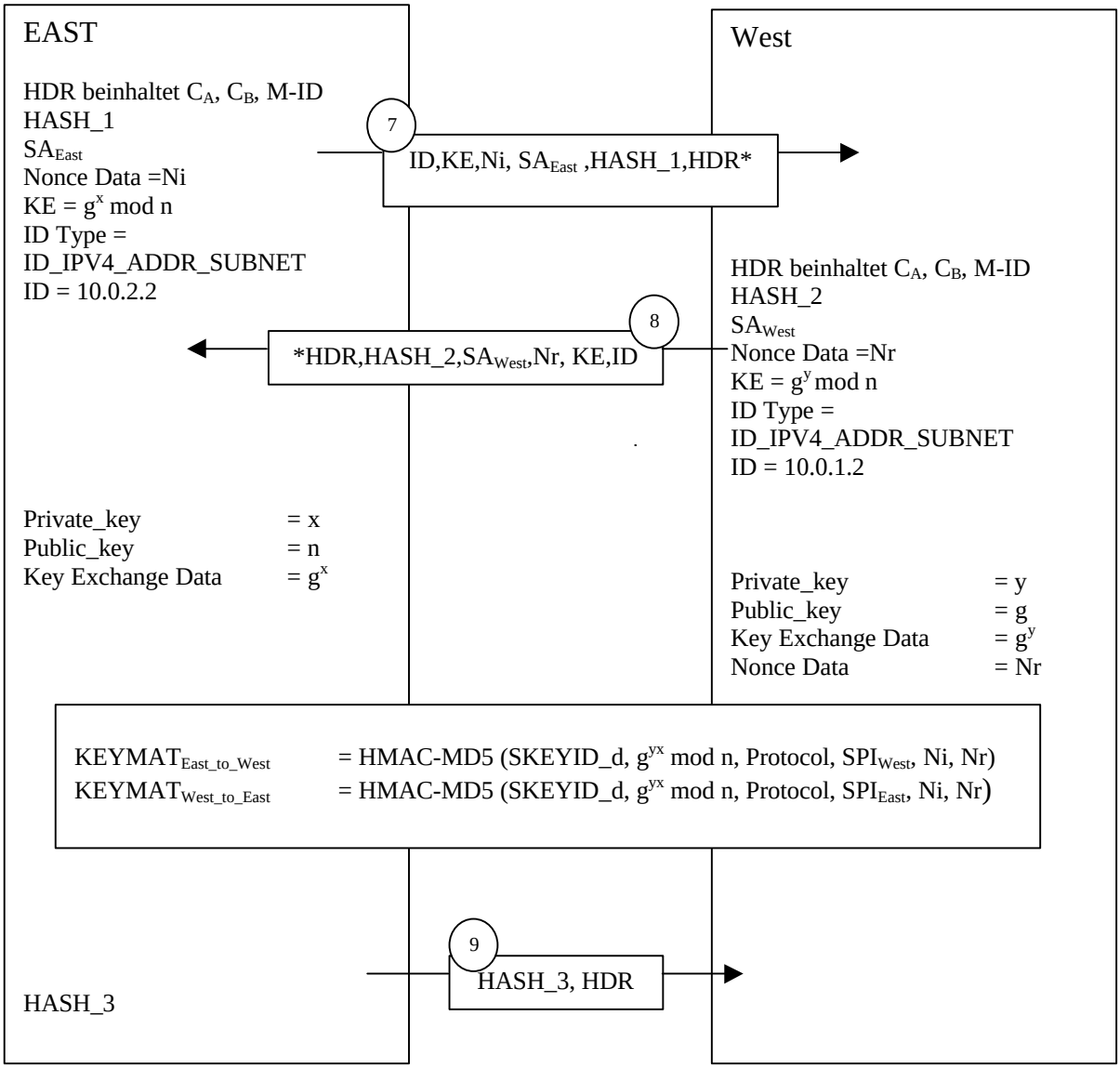
M_ID: Message Identität, da innerhalb einem Main Modus zwischen zwei Security Gateways mehrere Quick Modes existieren können, dient der M_ID zur Unterscheidung der Quick Modes

SA: Dabei muss gesagt, dass in der Regel die SA vom Initiator (‘East‘) umfangreicher ist als die SA von Responder (‘West’)

Ni, Nr: Zufallszahlen, welche für die Authentifizierung- und Schlüsselberechnung verwendet werden.

KE Key Exange Data, hier steht das Resultat der Diffie-Hellman Rechenoperation (g^x mod n) für $KE_{Initiator}$ und (g^y mod n) für $KE_{Responder}$.

7.7.1 Prozessbeschreibung des Quick Mode



Datenpaket 1: Für die richtige Authentifizierung wird der HASH_1 (*Hashwert*) nach dem ISAKMP-Header mitgeschickt. Die SA_{East} besitzt ein Proposal-Payload mit mehreren Transform Payloads, welche das Angebot an Verschlüsselungs- und Authentifizierungsalgorithmen beinhalten, welche dann auf die Nutzdaten angewendet werden sollen. Ni und KE dienen zur Authentifikation des Pakets, damit auch der 'West' den HASH_1 Wert bilden und vergleichen kann. Auch in diesem Mode wird ein geheimer Schlüssel mittels Diffie-Hellman erzeugt. Dies ist nötig um *Perfect Forward Secrecy* gewährleisten zu können. Zur Identifikation des Host wird ihre IP-Adresse verwendet.

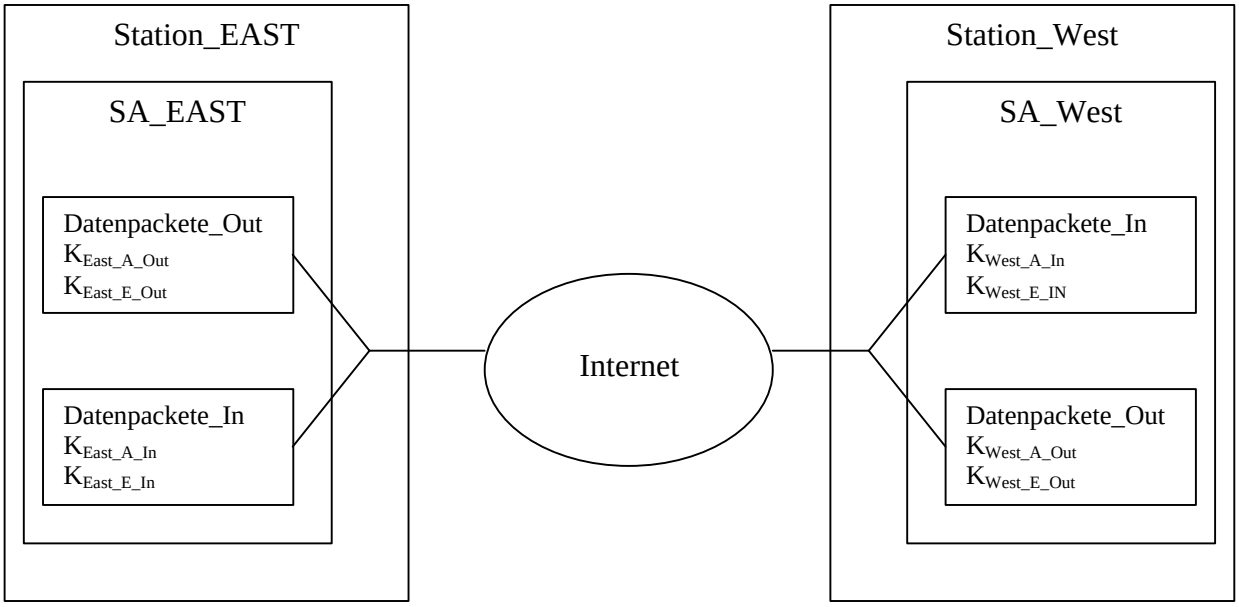
Datenpaket 2: SA_{West} beinhaltet das ausgewählte Tranform-Payload. Alles andere ist wie beim Datenpaket 1, natürlich mit anderen Parameter-Werten.

Ab diesem Zeitpunkt haben ‘West’ und ‘East’ alle Parameter, die es braucht um die Schlüssel zu generieren. Dabei muss man beachten, dass für ausgehende und hineinkommende Daten nicht die gleiche *SA* benützt werden. Es existieren also zwei *SA*’s für eine Verbindung. Jede Station muss also zwei Schlüsselpaare erzeugen:

$K_{Station\_A\_Out}$ für Authentifikationserzeugung für rausgehende Datenpakete
$K_{Station\_A\_In}$ für Authentifikationcheck für reinkommende Datenpakete

$K_{Station\_E\_Out}$ für Verschlüsselung von rausgehenden Datenpakete
$K_{Station\_E\_In}$ für Verschlüsselung von reinkommende Datenpakete

Die Schlüsselverteilung für eine Verbindung zwischen ‘East’ und ‘West’ sieht dann folgendermassen aus:



Geltende Beziehungen zwischen den Schlüsseln

$K_{East\_A\_Out} = K_{West\_A\_In}$
$K_{East\_E\_Out} = K_{West\_E\_IN}$
$K_{East\_A\_In} = K_{West\_A\_Out}$
$K_{East\_E\_In} = K_{West\_E\_Out}$

Datenpaket 3: Mit diesem Datenpaket bestätigt ‘West’ zu ‘East’ seine Bereitschaft Nutzdaten anhand der ausgehandelten SA (Datenpakete 7/8) zu behandeln. Somit steht einer sicheren Verbindung zwischen ‘East’ und ‘West’ übers Internet nichts mehr im Weg.

7.8 Angriffe im Netz

Wie wichtig die Informationen überhaupt sind, die über ein Netz geschickt werden, wird einem wahrscheinlich erst richtig bewusst, wenn für die Eingabe der eigenen Kreditkartennummer aufgefordert wird. Denn wenn jemand in den Besitz dieser Informationen kommt kann er ihnen durch Missbrauch einigen Schaden zufügen. Noch extremer ist die Situation bei Informationsaustausch innerhalb einer Firma. Die Konkurrenz könnte sich durch ergattern dieser Daten einen Wettbewerbsvorsprung verschaffen. Um Massnahmen gegen solche Lauschangriffe treffen zu können, muss man sich erst über die verschiedenen Angriff-Methoden im Klaren sein.

Sniffing

Die sogenannten Snifferprogramme dienen normalerweise der Fehlersuche und Überwachung des Netzverkehrs. Mit ihnen können Datenpakete und deren Inhalt innerhalb des gleichen Netzes mit Leichtigkeit analysiert werden.

Verweigerung von Diensten (Denial-of-Service-Angriff)

Das Ziel eines solchen Angriffs ist durch wiederholtes Versenden von ungültigen Paketen einen entfernten Rechner lahmzulegen. Dieser destruktive Akt wird gemacht um einer Firma mit dem temporären Ausfall des Netzwerks zu schaden.

Vortäuschen einer falschen IP-Adresse

In paketvermittelten Netzen ist es sehr schwierig nachzuweisen, welchen Weg ein Paket von Punkt A zu Punkt B durchläuft. Darum hat ein Angreifer die Möglichkeit durch Ändern seiner IP-Adresse einen anderen Ursprungsort vorzutäuschen. Diese Art von Angriff kann jede Menge schaden anrichten.

Man-in-the-Middle

Bei einem Man-in-the-Middle-Angriff sitzt der Angreifer zwischen zwei Teilnehmern A und B. Er nimmt zum einen die Daten von A auf, manipuliert sie und gibt sie an B weiter. Da er sich durch Vortäuschen einer falschen IP-Adresse als B ausgibt, erhält er auch die Antworten von B zurück.

Replay-Angriff

Der Angreifer zeichnet die Nachricht zwischen zwei Teilnehmern auf, speichert sie und sendet sie zu einem späteren Zeitpunkt erneut.

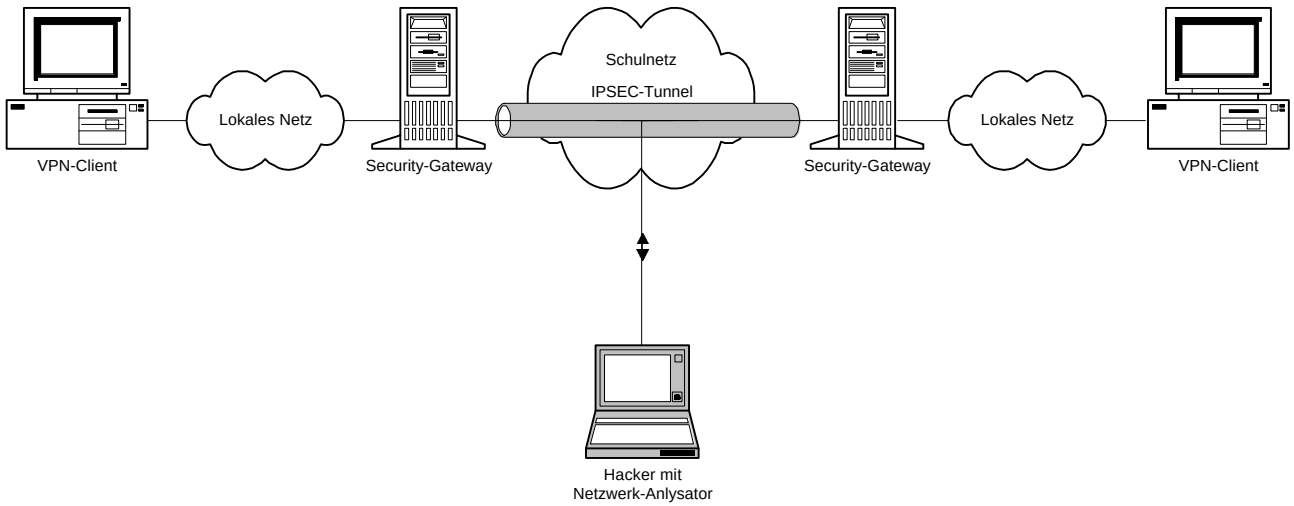
In VPN’s wird dieser Gefahr durch verschieden Methoden begegnet:

- Serialisierung der Pakete durch ID-Nummer
- Kapselung des IP-Headers
- Synchronisierte Timestamps
- Zeitabhängige Authentifizierung
- Periodischer Schlüsselwechsel

7.9 Testen der Sicherheit

Um die Sicherheit von IPSec zu verifizieren benötigt man einen Netzwerkanalyzer (Sniffer). Mit diesem Werkzeug kann man die einzelnen Datenpakete auf einem Netz aufnehmen, speichern, analysieren, filtern und sogar verändern und wieder auf das Netz einspeisen. Daher erweist sich dieses Tool als das ideale Werkzeug für die Standfestigkeit von IPSec gegen Eindringlinge zu testen.
In einer ersten Phase wurden vor allem die Datenpakete in verschlüsselter und unverschlüsselter Form analysiert. In der zweiten Phase wurde ein Replay-Angriff durch Senden von gespeicherten Paketen gemacht.

Versuchsanordnung



7.10 Paketanalyse

Der Netzwerkanalyzer ist über ein Hub am gleichen Netz wie die beiden Security-Gateways angeschlossen. Damit gezielt die Pakete aufgenommen werden, welche zwischen den beiden Gateways ausgetauscht werden, wird ein Filter auf die beiden entsprechenden IP-Adressen konfiguriert.
Normalerweise wird beim Übertragen von Dateien mit TCP/IP-Protokoll viel Verkehr verursacht, da einzelne Pakete immer wieder bestätigt werden. Der Einfachheit halber wurde deshalb das Steuerprotokoll *ICMP* (Internet Control Message Protocol) für diese Testzwecke gewählt. Mit dem Ping-Befehl wird eine *ICMP*-Nachricht (ECHO) an eine bestimmte Adresse geschickt. Bei Erhalt der Nachricht wird vom Ziel erwartet, dass es eine Nachricht (ECHO REPLAY) zurückschickt. Diese *ICMP*-Nachrichten sind immer in IP-Pakete verkapselt. Um den Inhalt und Ablauf zu definieren, wird der Text-String "Das ist ein Test" vom Client 'Sunrise' zum Client 'Sunset' übertragen. Der Befehl dazu sieht folgendermassen aus:

```
ping -c 1 -p 44617320697374206569 6E2054657374
```

Der Parameter -c 1 beschränkt die Ausführung auf einen einmaligen Vorgang und -p erlaubt die Übermittlung des Strings in Hex-Code.

Anzeige des Ping-Datenpakets beim Sniffer:

Number	DeltaTime	Destination	Source	Interpretation
1	0 sec	**10.0.1.2**	**10.0.2.2**	ICMP Echo Total_Len=84
2	260 us	**10.0.2.2**	**10.0.1.2**	ICMP Echo Reply Total_Len=84

Zur Analyse wird das vorerst noch unverschlüsselte Datenpaket Number 1 in Hex dargestellt:

```
0000:  08 00 2B C3 CF F2 08 00 - 2B C3 E6 CC 08 00 45 00  .+CO .+CfL .E.
0010:  00 54 02 A2 00 00 3F 01 - 62 04 0A 00 02 02 0A 00  .T "..? b .
0020:  01 02 08 00 94 80 B4 01 - 00 00 37 EA 45 33 00 0B    .4 ..7jE3.
0030:  A0 E9 44 61 73 20 69 73 - 74 20 65 69 6E 20 54 65   iDas ist ein Te
0040:  73 74 44 61 73 20 69 73 - 74 20 65 69 6E 20 54 65  stDas ist ein Te
0050:  73 74 44 61 73 20 69 73 - 74 20 65 69 6E 20 54 65  stDas ist ein Te
0060:  73 74                                              st....
```

Um dieses Datenpaket mit seinen Hex-Zahlen besser zu verstehen, wird hier näher auf den Aufbau des Datenpakets eingegangen. In diesem Fall besteht das Ping-Datenpaket aus 3 Protokollen, nämlich aus dem Ethernet, IP und *ICMP*. Jedes Protokoll kommt in Form seines Headers und dem nachfolgenden Nutzdatenteil vor. Zuerst kommt der Ethernet Header, wobei in seinen Nutzdaten das IP-Protokoll transportiert wird. Das heisst, dass nach dem Ethernet Header der IP-Header folgt mit seinen IP-Nutdaten. Da es sich hier um einen Ping-Befehl handelt, befindet sich in den IP-Nutzdaten
das ICMP-Protokoll, das wiederum mit einem ICMP-Header beginnt und diesmal in seinen Nutzdaten die Information besitzt, welche wir im Ping Befehl explizit mit ‘–p {Hex-String}‘ angegeben haben. Der Grund, wieso der eingegebene Hex-String dreimal vorkommt ist, weil das gesendete Datenpaket im IP-Nutzdatenfeld die bestimmte Minimallänge von (64KByte) haben muss. Dies ist nötig, um Kollisionen auf dem Ethernet-Übertragungsmedium sicher zu dedektieren.

```
0000:  ++++++++++++Ethernet-Header++++++++++ #####  .+CO .+CfL .E.
0010:  ##################IP-Header###################  .T "..? b .
0020:  #### ************************************    .4 ..7jE3.
0030:  ***************ICMP-Header*****************   iDas ist ein Te
0040:  ***************ICMP-Nutzdaten**************  stDas ist ein Te
0050:  ****************************************** stDas ist ein Te
0060:  ****                                        st....
```

Die ersten 12 Byte geben die MAC-Destination-Adresse (08 00 2B C3 CF F2) and MAC-Source-Adresse (08 00 2B C3 E6 CC) an. Die nächsten zwei Byte (08 00) weisen darauf hin, dass ein IP-Paket daran anschliesst. Danach folgt der IP-Header mit 20 Byte. Am Schluss des IP-Headers sieht man die IP-Source-Adresse (0A 00 02 02) und IP-Destination-Adresse (0A 00 01 02) , bevor die IP-Daten folgen. In diesem Fall beinhalten die IP-Daten das *ICMP*-Paket. Beim ICMP-Header wird mit dem ersten Byte (08 entspricht Echo-Request) der Typ des ICMP-Pakets angegeben, danach folgen die ICMP-Daten.
Jetzt wird ein verschlüsseltes Datenpaket näher analysiert. Der Netzwerkanalyzer nimmt den einmaligen Ping Befehl folgendermassen auf:

Number	DeltaTime	Destination	Source	Interpretation
1	0 sec	**160.85.131.60**	**160.85.131.61**	IP **ID=78** Protocol=Unknown FramentOffset=0 Total_Len=136
2	1.1 ms	**160.85.131.61**	**160.85.131.60**	IP **ID=31** Protocol=Unknown FramentOffset=0 Total_Len=136

Unten sehen wir das verschlüsselte Datenpaket Nummer 1 in Hex-Form näher an:

```
0000:  08 00 2B C3 CF F2 08 00 - 2B C3 E6 CC 08 00 45 00  .+CO .+CfL .E.
0010:  00 88 01 FB 00 00 40 32 - 31 25 A0 55 83 3D A0 55  . {..@21% U = U
0020:  83 3C FE B9 ED 64 00 00 - 00 01 DA CC 55 0B D6 8E  <~9md.. ZLUV
0030:  90 BB 3C 79 00 EA 6A 68 - EA 98 62 31 CF 26 4D EC  ;<y.jjh b1O&Ml
0040:  E5 F1 EA 83 4F 43 29 69 - D7 AD 6F B3 BC A4 33 C0 eqj OC)iW-o3<$3@
0050:  E1 96 87 D2 0A 97 8D 58 - 64 04 E3 4A D7 24 35 2E a R .Xd cJW$5.
0060:  4B 3B A5 09 12 A7 53 84 - A7 4D 11 F8 DE 8A FD 03 K;%. 'S 'M x^}
0070:  19 BB 38 EA 5F 13 FE 44 - 21 46 21 79 0A 0C 73 72  ;8j ~D!F!y sr
0080:  81 8F 5B 1D 49 86 40 88 - 3E 6D CC ED 15 54 BA 91   I @ >mLm T:
0090:  B6 80 8E 35 EB 8E 00 00 - 00 00                    6.5k....
```

Auf den ersten Blick fällt natürlich sofort auf , dass dasselbe Paket insgesamt (ca. 60%) grösser geworden ist und das der eingegebene String nicht mehr in lesbarer Form vorliegt. Grund dafür ist die Verschlüsselung, welche mittels ESP im Tunnelmodus vorgenommen worden ist. Durch die Verschlüsselung wird die vollumfängliche Analyse des Datenpakets erschwert. Was aber sofort ins Auge sticht, ist die Anwendung des ‘Tunneling‘. MAC- und IP-Adressen des ursprünglichen Datenpakets sind jetzt verschlüsselt in den Nutzdaten.Was man jetzt sieht, sind die Adressen der MAC- und IP-Adressen der Security-Gateways. Somit ist die Identität der eigentlichen Kommunikationspartner im Verborgenen.
Ausserdem sind auch die Daten nicht mehr als ICMP-Nachricht erkennbar, denn diese befindet sich jetzt in verschlüsselter Form in den Nutzdaten des neuen IP Headers, welcher mit ESP bearbeitet worden ist. Im Kapitel ESP [→ ESP] ist beschrieben, wie ein Datenpaket im Tunnelmodus mit ESP aussieht und wie die ESP-Header aufgebaut sind. Wegen den variablen Längenangaben der Datenfelder innerhalb des ESP Bereiches kann nur teilweise über den Aufbau Auskunft gegeben werden. Bezogen auf diese Informationen können wir sagen, dass die ersten 8 Byte (SPI und Sequence Number) zum ESP-Header gehören und nicht verschlüsselt übertragen werden. Die letzten 12 Byte gehören dem Authenfifizierungscode, denn wie wir wissen, wird vom erzeugten *Hashwert* nur die 96 höchstwertigsten Bits verwendet. Zwischen ESP-Header und ESP-Auth befinden sich die verschlüsselten Daten des ursprünglichen Datenpakets.

7.11 Replay-Angriff

Der Netzwerk-Analyzer liest die Datenpakete, welche zwischen den beiden Security-Gateways verlaufen, und speichert sie. Somit können diese Pakete zu einem beliebigen Zeitpunkt unverändert nochmals abgeschickt werden. Es wird wiederum die selbe *ICMP*-Nachricht [→ ICMP] von Client Sunrise nach Client Sunset gesendet. Bei Sunset wird zudem mit Hilfe des Sniffer-Programms `TCPdump` der Datenverkehr aufgezeichnet, um sicher zu stellen, dass er das Request-Signal erhält und quittiert.
Uns interessiert in erster Linie, inwiefern sich Datenpakete in verschlüsselter und unverschlüsselter Form manipulieren lassen und wie sicher die Abwehrfunktionen von IPSec sind. Nachfolgend sind verschiedene Tests und ihre Resultate beschrieben.

1. Ein unverschlüsseltes *ICMP*-Paket, welches für den Client Sunrise bestimmt ist, wird unverändert wieder abgeschickt.
   Resultat: Das Paket wird durch den Gateway geroutet, erreicht den Client Rechner und wird mit ECHO REPLAY quittiert.

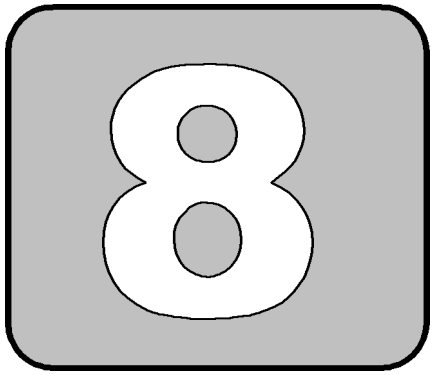
2. Nun wird ein Zeichen in den Nutzdaten des Pakets verändert.
   Resultat: Das Paket erreicht zwar den Client wird dort aber wegen „Invalid Checksum“ verworfen.

3. Ein zweites Zeichen wird so verändert, dass die Checksumme wieder stimmt.
   Resultat: Das Paket erreicht den Client und wird sogar quittiert. Der Client wurde überlistet!

4. Nun werden gleiche aber verschlüsselte Datenpakete aus dem VPN-Tunnel genommen, gespeichert und unverändert wieder abgeschickt.
   Resultat: Das Paket kommt bis zum Security-Gateway und wird bereits von ihm verworfen. Weitere Test mit Manipulationen sind nicht mehr sinnvoll, da selbst unveränderte Pakete als Wiederholung erkannt werden.

Erkenntnis

Dieser Versuch illustriert die Resistenz von IPSec gegen Replay-Angriffe. Die Datenpakete werden mit einer Folgenummer versehen. Diese ermöglicht die Erkennung einer Mehrfachübertragung. In einem solchen Fall verwirft der Gateway das Paket.

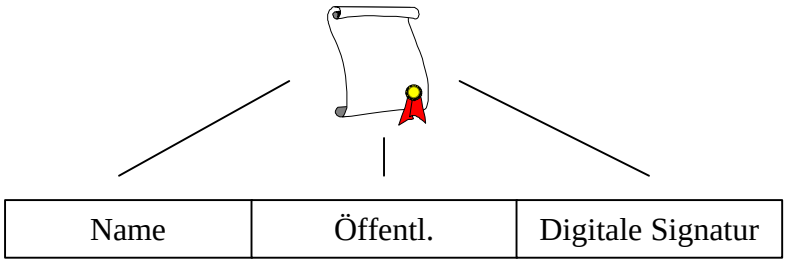


8 Zertifikate

In diesem Kapitel werden Sinn und Zweck von Public-Key-Zertifikaten und deren Verwendung beschrieben. Weiter wird demonstriert wie die Verwendung von Zertifikaten mit IPSec realisiert wird.

8.1 Was ist ein Zertifikat ?

Ein Zertifikat hat die Aufgabe zu bestätigen, dass ein bestimmter öffentlicher Schlüssel (Public Key) zu einem bestimmten Namen (Identität) gehört. Heute werden sog. Client-Zertifikate in Browser und E-Mail-Programmen eingesetzt, um sich gegenüber Dritten ausweisen zu können. Server-Zertifikate dienen dazu, dass sich ein Server gegenüber Benutzern identifizieren, und somit verschlüsselt mit SSL kommunizieren kann. Sie sind eine Lösung um gewisse Sicherheitslücken im Internet zu schliessen. Zertifikate enthalten in jedem Fall den Namen des Inhabers des zugehörigen privaten Schlüssels, einen öffentlichen Schlüssel und eine digitale Signatur, welche die Echtheit bestätigt.



Ein Beispiel:
Nehmen wir an, sie möchten in einem Shop etwas kaufen und mit der Kreditkarte bezahlen. Der Verkäufer will sicher sein, dass die Karte gültig ist und überprüft dazu folgende Sachen:

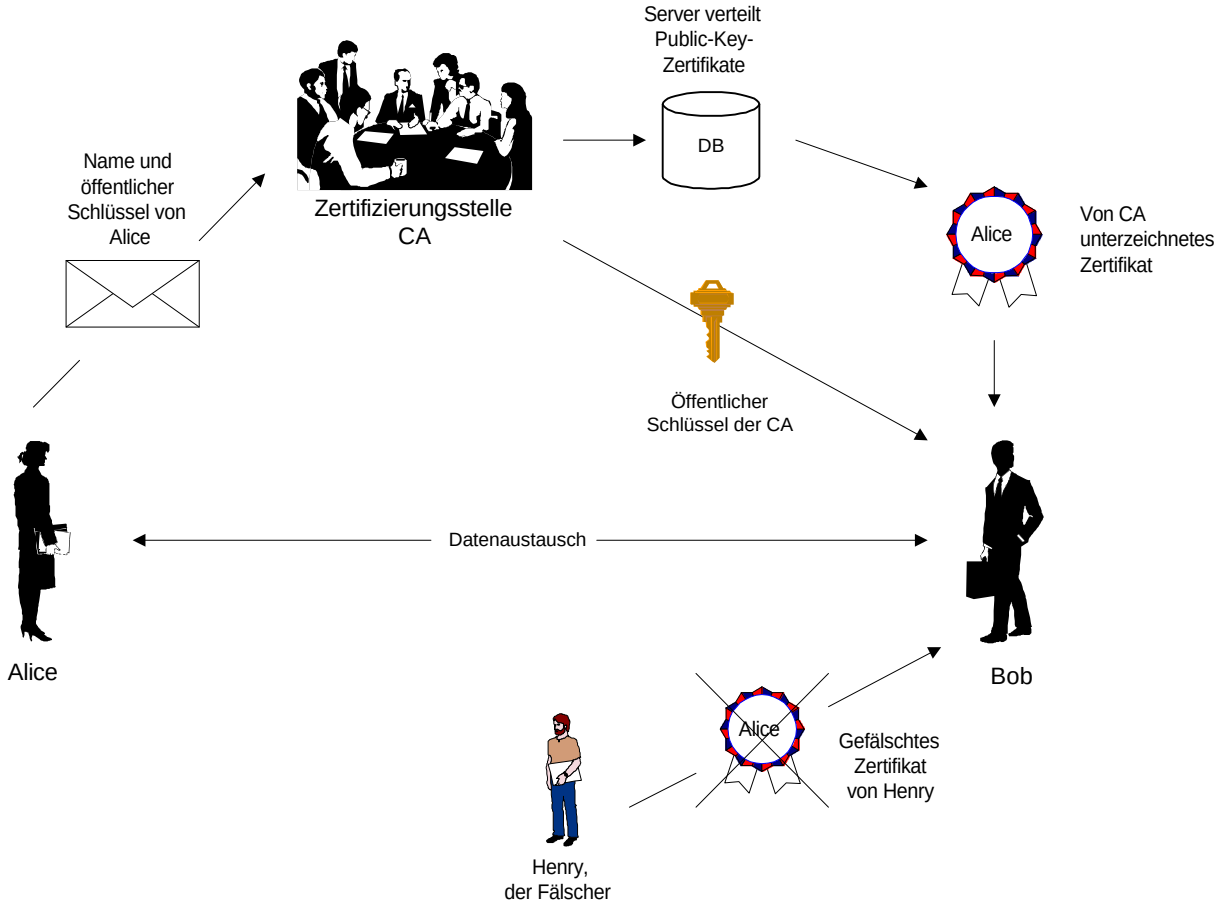
1. Anhand dem schwer kopierbaren Hologramm auf der Karte kann er feststellen, ob es sich um eine Fälschung handelt oder nicht.
2. Der Verkäufer überprüft, ob die Unterschrift des Inhabers nachträglich verändert wurde.
3. Ist die Karte von einer Kreditkartenfirma (VISA, Eurocard) ausgestellt worden?
4. Ist die Karte überhaupt noch gültig? Bei Kreditüberschreitungen oder nach Ablauf werden Karten ungültig. Dazu benützt er den Kartenleser, welcher sich bei der Kartenzentrale über die Gültigkeit erkundigt.
5. Die Identität des Karteninhabers wird bestätigt, indem er eine Unterschrift macht. Der Verkäufer vergleicht dann die Unterschrift mit jener auf der Karte.

Das digitale Zertifikat ist ein elektronisches Dokument, das analog zur Kreditkartenüberprüfung die selben Eigenschaften aufweist. Es enthält die Identität einer Person (Personalien), seinen öffentlichen Schlüssel und die digitale Signatur einer vertrauenswürdigen Organisation.

1. Anhand der digitalen Signatur kann eine Fälschung ausgeschlossen werden.
2. Zur Kontrolle einer nachträglichen Veränderung der Unterschrift dient ein Hashwert.
3. Zertifikate werden von einer vertrauenswürdigen Zertifizierungsstelle ausgestellt.
4. Zertifikate, welche nicht mehr gültig sind, werden in die CRL (Certificate Revocation List) eingetragen und von der Zertifizierungsstelle publiziert.
5. Eine Identität kann durch eine dritte Partei bestätigt werden, indem eine zufällig generierte Nachricht, die mit dem privaten Schlüssel verschlüsselt worden ist, durch Entschlüsselung mit dem im Zertifikat enthaltenen öffentlichen Schlüssel wieder der ursprünglichen Nachricht entspricht.

8.2 Verteilung öffentlicher Schlüssel

Zertifikate werden von einer Zertifizierungsstelle (Certification Authority CA) unterzeichnet. Sie gilt als vertrauenswürdige dritte Partei. Wenn man im Besitz des öffentlichen Schlüssels der CA ist, kann man damit alle Zertifikate überprüfen, welche von der CA unterzeichnet wurden.

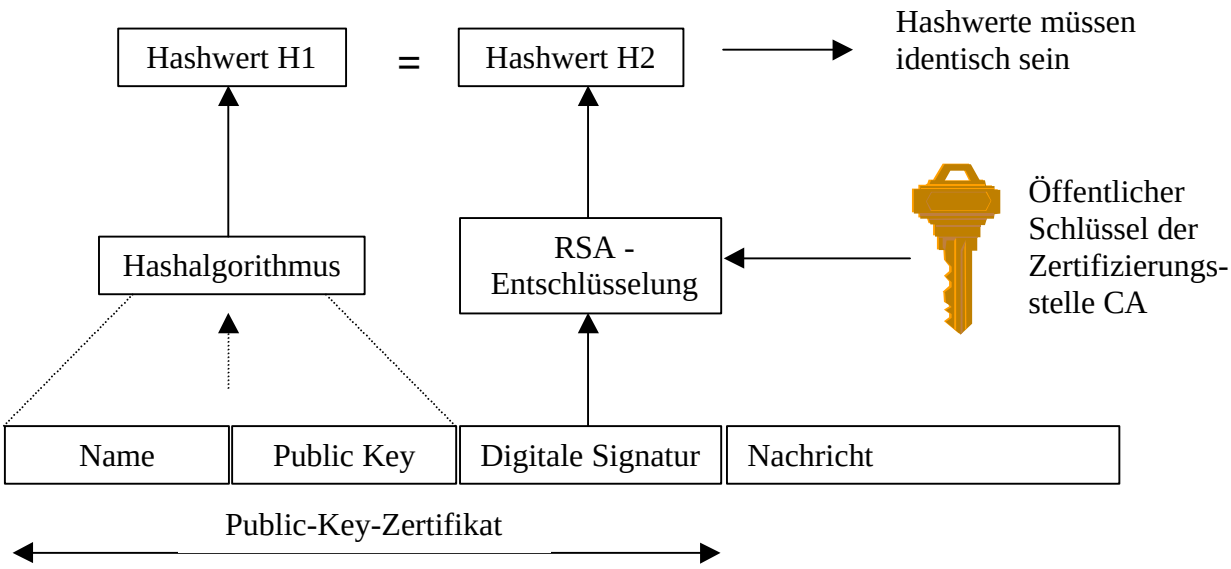


Angenommen Alice möchte Bob eine Kopie ihres öffentlichen Schlüssels senden, so wendet sie sich zuerst an die CA. Diese unterzeichnet Alice's Zertifikat. Ab diesem Zeitpunkt beinhaltendas Alice nun zuverlässig an Bob senden kann. Sowohl Alice als auch Bob vertrauen der CA. Nach dem Empfang kann Bob die Echtheit des Zertifikats überprüfen. Dazu benötigt er den öffentlichen Schlüssel der CA.
Grundsätzlich gibt es zwei Konzepte zur Erzeugung von Public-Key-Schlüsselpaaren.

1. Auf dem Rechner des Schlüsselinhabers werden die Schlüsselpaare erzeugt. Den privaten Schlüssel behält der Inhaber und schickt den öffentlichen Schlüssel zur Erstellung des Zertifikats an die CA.
   Vorteil: Private Schlüssel müssen nicht kopiert werden (bleibt beim Inhaber).
   Nachteil: Öffentliche Schlüssel müssen an die CA gesendet werden (aufwendig).

2. Das Schlüsselpaar und das unterzeichnete Zertifikat werden von der CA erzeugt und an den Benutzer geschickt.
   Vorteil: Die Schlüsselerzeugung kann von mehreren Benutzer gemeinsam genutzt werden.
   Nachteil: Private Schlüssel müssen kopiert und zugestellt werden (Sicherheitsrisiko).

8.3 Überprüfung des Zertifikats



Die CA versieht das Zertifikat bei einer Unterzeichnung mit einer digitalen Signatur. Das ist nichts anderes als der *Hashwert* chiffriert mit dem privaten Schlüssel. Der Hashwert H1 wird vom öffentlichen Schlüssel (Public Key) und vom Namen des Inhabers gebildet, denn es will ja deren Zusammengehörigkeit bestätigt werden. Zur Überprüfung des Zertifikats kann Bob die digitale Signatur mit dem öffentlichen Schlüssel der CA wieder dechiffrieren und erhält somit wieder den Hashwert H2. Dieser vergleicht er mit dem selber berechneten Hashwert H1. Stimmen die beiden Hashwerte überein, so ist das Zertifikat gültig. Henry der Fälscher hat also keine Chance mit einem gefälschten Zertifikat Bob zu täuschen, da er es nicht mit dem privaten Schlüssel der CA unterzeichnen konnte. Sobald Bob im Besitz von Alice's echten öffentlichem Schlüssel ist, können Daten ausgetauscht, entschlüsselt und authentisiert werden.

8.4 Public-Key Infrastruktur (PKI)

Die Zertifizierungsstelle ist in der Regel eine zentrale Instanz, welche öffentliche Schlüssel verwaltet und deren Echtheit garantiert. Nun stellt sich aber die Frage erneut nach der Echtheit der CA. Es könnte ja sein, dass sich jemand anders als CA ausgibt. Wie kann man der CA vertrauen?
Dafür gibt es verschiedene Strukturen von Zertifikatssystemen die sog. PKI (Public-Key-Infrasructure). Allgemein sind sie in drei Kategorien einteilbar:

1. **Zentrale Zertifizierungsstelle**. Alle Zertifikate werden von einer einzigen CA unterzeichnet und meist manuell an die Hosts verteilt.
2. **Hierarchische Zertifizierung**. Die Zertifikate der lokalen CA's werden von übergeordneten Instanzen unterzeichnet. An der Spitze der Hierarchie steht in der Regel eine anerkannte Institution (Root CA).
3. **Vertrauensgeflecht (Web of Trust)**. Jeder Hosts kann sich selber als CA betätigen und Zertifikate unterzeichnen. Darum liegt es am Benutzer zu entscheiden, wem er vertrauen will und wem nicht. Dieses Konzept wird z.B. in Pretty Good Privacy (PGP) verwendet.

8.5 Authentisierung mit Zertifikaten bei IPSec

Um Authentisierung mit Zertifikaten bei IPSec anzuwenden sind einige Modifikationen notwendig. In den folgenden Schritten wird beschrieben, welche Komponenten dazu installiert und konfiguriert werden müssen.

1. Zuerst muss das Modul OpenSSL installiert werden. OpenSSL stellt dem Betriebssystem Bibliotheken zur Verfügung, welche die Protokolle Secure Socket Layer (SSL v2/v3) und Transport Layer Security (TLS v1) implementiert. Zudem werden verschiedene Verschlüsselungsarten sowie die Erstellung von Zertifikaten unterstützt

2. In einem nächsten Schritt wird das bereits installierte IPSec modifiziert. Dies geschieht mit Hilfe von einem Patch. Dieser sucht gewisse Einträge im Quellcode und ersetzt diese durch neue Einträge. Danach muss ein neuer Kernel neu kompiliert werden.

3. Nun können die Zerifikate hergestellt werden.

Die Bezugsquelle der beiden benötigten Dateien finden sie im Quellenverzeichnis:

[4] OpenSSL Modul
[3] Patch für FreeS/WAN (IPSec)

8.5.1 Installation von OpenSSL

Um sicher zu gehen, dass OpenSSL in einem definierten Pfad installiert wird, lohnt es sich den Pfad explizit anzugeben:

```
./config --prefix=/usr/local/ssl --openssldir=/usr/local/ssl
```

Danach müssen folgende Befehle der Reihe nach ausgeführt werden:

`make` Die Biblitheken werden übersetzt:

`make test` Nun kann der letzte Vorgang getestet werden. Sollten hier Fehlermeldungen erscheinen, empfiehlt es sich die Dateien mit *`make clean`* zu entfernen und die Bibliotheken mit *`make`* nochmals zu übersetzen.

`make install` Nun werden im OpenSSL-Stammverzeichnis die folgenden Unterverzeichnisse erstellt:

/certs
/misc
/private
/bin
/include
/lib

8.6 Modifikation von IPSec

Um eine Authentisierung mit Hilfe von Zertifikaten machen zu können, müssen ein paar IPSec-Dateien modifiziert werden. Dies geschieht im Allgemeinen mit einem Patch. Da IPSec für den Transport der Zertifikate das Secure-Socket-Layer-Protokoll benötigt, muss zuerst OpenSSL installiert werden.
[→ Installation von OpenSSL]
Die komprimierte Patch-Datei heisst **`pluto-openssl.tar.gz`** und enthält drei Dateien:

-README.certificates	Wertvolle Anleitung zur Installation und Erzeugung von Zertifikaten
-pluto.diff	Enthält die Änderungen für Dateien im Pluto-Verzeichnis
-utils.diff	Enthält die Änderungen für Dateien im Utils-Verzeichnis

Diese Datei muss zuerst im Stammverzeichnis des IPSec *(/usr/src/freeswan-snap1999Sep16b*) entpackt werden. Danach werden mit dem Patch-Befehl die Modifizierung vorgenommen.

```
patch –p0 < pluto.diff
patch –p0 < utils.diff
```

Weil sich die Snapshot-Versionen laufend verändern, ist es durchaus möglich, dass während diesem Vorgang Fehlermeldungen erscheinen. In diesem Fall konnte der Originaltext für die Modifikation nicht eindeutig lokalisiert werden. Von den betroffenen Dateien werden dann sog. Reject-Files (*.rej) erzeugt. Sie enthalten die fehlgeschlagenen Änderungen. Anhand von diesen Informationen kann man der Ursache nachgehen und gegebenenfalls die nötigen Einträge manuell vornehmen.

Damit Pluto das Stammverzeichnis von OpenSSL kennt, muss die Datei *Makefile* im *Pluto*-Verzeichnis editiert werden und ein Eintrag geändert werden. Es handelt sich um folgende Zeile:

```
OPENSSLROOT= /usr/local/ssl
```

Danach muss FreeS/WAN neu installiert werden, indem ein neuer Kernel erstellt wird. Dieser Vorgang wird im Kapitel [→ Installation von FreeS/Wan] ausführlich beschrieben.

8.7 Die Erzeugung von Zertifikaten

Ein Zertifikat ist im Grunde genommen nichts anderes als eine Beglaubigung. Es basiert auf dem Vertrauen in eine dritte Partei. Diese dritte Partei ist die Zertifizierungsstelle (Certification Authority CA). Sie versieht die Datensätze der Teilnehmer mit einer digitalen Signatur (Hashwert mit privatem Schlüssel kodiert) und gibt sie auf Anfrage an jeden Kommunikationspartner weiter. Die ganze erforderliche Architektur nennt man auf Neudeutsch auch PKI (Public Key Infrastructure). Der dynamische Austausch von Zertifikaten ist mit FreeS/WAN leider noch nicht möglich. Deshalb wird in einem ersten Schritt auf einem Rechner eine zentrale Zertifizierungsstelle eingerichtet. Dieser Rechner muss sich nicht unbedingt im lokalen Netz befinden.
Die Kommunikationsteilnehmer (Hosts) stellen einen Antrag an die CA, indem sie eine Request-Datei erzeugen. Diese wird von der CA signiert und an alle Hosts verteilt. Der Transport der Zertifikate erfolgt manuell z.B. durch Kopieren auf Diskette.

8.7.1 Einrichten der Zertifizierungsstelle

Bevor mit dem eigentlichen Einrichten des CA-Servers begonnen werden kann, müssen noch ein paar Vorbereitungen getroffen werden. In der Datei */usr/local/ssl/openssl.cnf* sollte sichergestellt sein, dass die Verzeichniseinträge im Abschnitt CA_default wie folgt aussehen:

```
[ CA_default ]

dir           = /usr/local/ssl          # Where everything is kept
certs         = $dir/certs              # Where the issued certs are kept
crl_dir       = $dir/crl                # Where the issued crl are kept
database      = $dir/index.txt          # database index file.
new_certs_dir= $dir/newcerts            # default place for new certs.

certificate  = $dir/cacert.pem          # The CA certificate
serial       = $dir/serial              # The current serial number
crl          = $dir/crl.pem             # The current CRL
private_key  = $dir/private/cakey.pem   # The private key
RANDFILE     = $dir/private/.rand       # private random number file
```

Weiter soll im *openssl* Stammverzeichnis eine Textdatei *ca.ext* mit folgenden Inhalt erstellt werden:

```
# Extensions for a typical CA - PKIX recommendation.

SubjectKeyIdentifier      = hash
AuthorityKeyIdentifier    = keyid:always,issuer:always
basicConstraints          = CA:true
keyUsage                  = cRLSign, keyCertSign
nsCertType                = sslCA, emailCA
subjectAltName            = email:copy
```

Im Stammverzeichnis /usr/src/ssl wird nun das CA-Zertifikat erstellt.

```
openssl req –new –newkey rsa:1024 \
 –keyout /usr/local/ssl/private/cakey.pem –out careq.pem
```

Normalerweise lassen sich die CA's ihre Zertifikate von einer übergeordneten Instanz unterzeichnen. Ist die CA bereits höchste Instanz, so muss sie sich ihr Zertifikate selbst signieren:

```
openssl x509 –CAcreateserial –signkey private/cakey.pem –req \
 -in careq.pem –out cacert.pem –days 2000 –extfile ca.ext
```

Dieses Zertifikat kann nun benutzt werden um weitere Zertifikate zu signieren.
Es kann nun eine Datenbank erzeugt werden, welche alle ausgestellten Zertifikate verwaltet:

```
touch index.txt
echo "01" > serial
```

8.7.2 Erstellen der CRL

Die Zertifizierungsstelle hat auch die Kontrolle über die Gültigkeit der Zertifikate. Sie kann, sei es aus Befristung oder Missbrauch, ein Zertifikat löschen. Die Information, dass ein Zertifikat nicht mehr gültig ist, wird in der CRL-Datei (Certificate Revocation List) gespeichert. Eine CRL-Datei wird folgendermassen generiert:

```
openssl ca –gencrl –out crl.pem
```

Damit alle Kommunikationspartner von der Gültigkeit der Zertifikate Kenntnis haben, muss die Datei crl.pem an alle Hosts verteilt werden.

8.7.3 Erstellen eines Host-Zertifikats

Damit die CA ein Zertifikat ausstellen kann, müssen ihr die Daten des jeweiligen Antragstellers bekannt sein. Für diesen Zweck muss der Host eine Request-Datei erzeugen, welche die Angaben wie seine IP-Adresse und sein Domain-Name enthält. Mit dem folgenden Befehl wird eine solche Request-Datei sowie ein privater und öffentlicher 1024-Bit-Schlüssel erzeugt (xxx = Hostname):

```
openssl req –new –newkey rsa:1024 \
      –nodes –keyout /etc/ipsec/xxx.key –out xxx.req.pem
```

Nach der Ausführung wird man aufgefordert, weitere Angaben wie Land, Adresse, Name der Organisation, E-Mail-Adresse und Passwort anzugeben. Diese Datei wird nun zum CA-Server transportiert. Dies sollte wiederum auf sicherem Weg passieren (z.B. manuell auf Floppydisk oder mit PGP verschlüsselter Mail).

8.7.4 Unterzeichnen des Zertifikatsantrags

Beim CA-Server muss vor jedem Signieren eines neuen Zertifikats ein Eintrag in der Datei openssl.cnf angepasst werden.

Im Abschnitt **[CA_default]**:

```
X509_extensions = svr_cert
```

Im Abschnitt [**svr_cert**]:

```
basicConstraints=CA:FALSE
nsCertType            = server
keyUsage = nonRepudiation, digitalSignature, keyEncipherment
nsComment             =    "IPsec certificate for xxx.zhwin.ch"
subjectKeyIdentifier=hash
authorityKeyIdentifier=keyid,issuer:always
subjectAltName=email:copy,DNS:xxx.zhwin.ch,IP:160.85.xx.xx
issuerAltName=issuer:copy
```

Des Zertifikat wird schliesslich mit folgendem Befehl signiert:

```
openssl ca –in xxx.req.pem
```

Das neue Zertifikat befindet sich nun im Verzeichnis */usr/local/ssl/newcerts* und kann von dort aus an diejenigen Hosts verteilt werden, die miteinander kommunizieren wollen. Sie werden lokal im Verzeichnis */etc/ipsec* gehalten.
Auf dem CA-Server kann danach die Request-Datei wieder gelöscht werden.

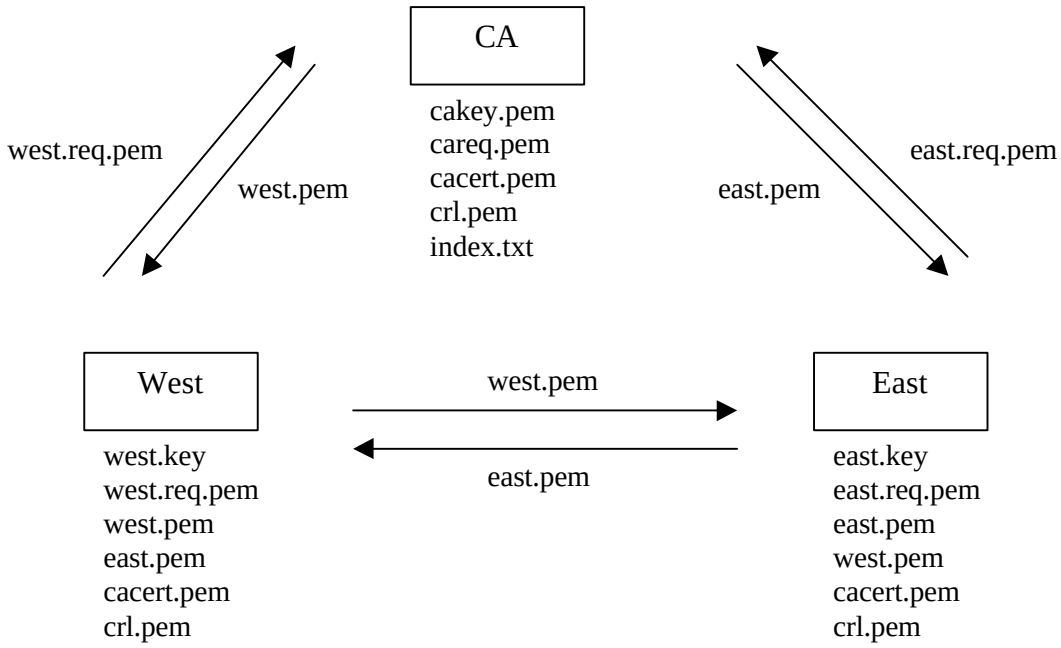
8.7.5 Widerrufen von Zertifikaten

Um die Gültigkeit eines Zertifikats zu widerrufen muss es aus der Datenbank der CA gelöscht werden. Dieses wird mit dem folgenden Befehl gemacht:

```
openssl ca –revoke xxx.pem
```

Wichtig: Damit alle Hosts Kenntnis von dem widerrufenen Zertifikat bekommen, muss wieder eine neue CRL-Liste erzeugt werden [→ Erstellen der CRL] und an alle Hosts verteilt werden.

8.7.6 Distribution der Zertifikate



Die Grafik zeigt auf, wie die verschiedenen Dateien in unserem VPN verteilt sind. Die beiden Security-Gateways 'East' und 'West' besitzen die Zertifikate für den Aufbau des IPSec-Tunnels.

8.8 Konfiguration von IPSec

Zur neuen Konfiguration von IPSec müssen in der Datei *etc/ipsec.conf* Einträge gemacht werden. Diese sind hauptsächlich für die verschiedenen Pfadangaben bestimmt. Es wird angegeben, in welchen Dateien sich die verschiedenen Zertifikate befinden. Es handelt sich um folgende Einträge:

```
#certfile
certfile=/etc/ipsec/ksy008.pem

#keyfile
keyfile=/etc/ipsec/ksy008.key

#peerfile
peerfile=/etc/ipsec/ksy009.pem

#certpath
cert-
path=/usr/local/ssl/cacert.pem:/usr/local/ssl/crl.pem:/usr/local/ssl/certs

#certopts
certopts=strict
```

Certfile: In dieser Datei befindet sich das eigene Zertifikat
Keyfile: Der private Schlüssel
Peerfile: Das Zertifikat des Kommunikationspartners
Certpath: In diesem Verzeichnis befinden sich die Zertifikate der CA und die CRL-Datei
Certopts: Hier können weitere Optionen zur Authentisierung und Verschlüsselung angegeben werden. Die meisten Optionen sind zwar vorgesehen aber noch nicht implementiert.

Es können mehrere Pfade oder Dateinamen angegeben werden, die aber durch einen Doppelpunkt getrennt werden müssen.

8.8.1 Umschalten auf Shared Secrets Authentifizierung

Um wieder zurück zur Authentisierung mit Pre-Shared-Secrets zu gelangen, müssen ein paar merkwürdige Kunstgriffe im Source-Code vorgenommen werden. Und zwar handelt es sich um die Datei *auto* im Verzeichnis */usr/local/lib/ipsec*. Die folgende Codezeile muss ersetzt werden:

Alt: **`if (certfile != " ")`** Neu: **`if (certfile != "- -certfileX")`**

In der Konfigurationsdatei *etc/ipsec.conf* muss auch noch angepasst werden. Gewisse Einträge müssen entfernt, andere stehengelassen werden. Nach Angaben des Autors des OpenSSL-Patches wird dies aber in einer neueren Version eleganter gelöst sein. Es handelt sich um folgende Einträge:

```
certfile=
keyfile=
peerfile=
#certpath
certopts=
```

Nun sollte wieder ein Verbindungsaufbau mit Pre-Shared-Secrets möglich sein.

8.9 Strict Mode

Der Strict Mode ist eine Option der Authentisierung, der als Eintrag in der IPSec-Kofigurationsdatei *ipsec.conf* unter *certopts* gesetzt werden kann. Ist der Strict Mode aktiviert, so werden die Bedingungen zur Überprüfung der Zertifikate viel genauer.
Wenn die CRL nicht vorhanden, falsch unterzeichnet oder abgelaufen ist, so wird die Überprüfung des Zertifikats fehlschlagen. Bei Vorhandensein der CRL wird die Liste durchgegangen. Das zu überprüfende Zertifikat darf sich nicht in der CRL befinden, denn das würde bedeuten, dass es sich um ein widerrufenes Zertifikat handelt.
Zum zweiten werden die in ISAKMP-Verfahren ausgetauschten ID-Daten mit den Angaben im Zertifikat verglichen. Konkret heisst das, dass die aktuelle Ipv4-Adresse des Kommunikationspartners mit der im Zertifikat unter 'Subject Alternativ Name' verankerten Angaben übereinstimmen müssen. 'Subject Alternativ Name' erthält zum Zeitpunkt der Unterzeichnung die entsprechende IP-Adresse und DNS-Name. Dies verursacht zum Beispiel bei Verwendung von dynamischen IP-Adressen Probleme [→ Remote Access].
Man muss sich also im Klaren sein, dass ein Verbindungsaufbau ohne Strict Mode, eine Reduktion der Sicherheit bedeutet.

8.9.1 Verbindungsaufbau

Sobald in *ipsec.conf* eine neue Verbindung konfiguriert ist, kann mit zwei IPSec-Kommandos eine Verbindung aufgebaut werden. Diese unterscheiden sich nicht vom Verbindungsaufbau mit *Shared Secrets*. Zuerst wird an den beiden Rechner, welche miteinander kommunizieren die Verbindung konfiguriert:

```
cd /usr/local/lib/ipsec
./ipsec auto –-show –-add {Verbindungname}
```

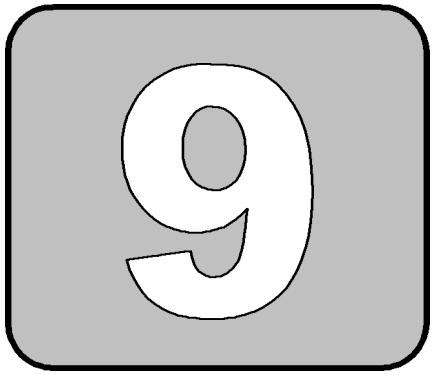
Und dann von einem der beiden Stationen die Verbindung gestartet:

```
./ipsec auto –-show –-up {Verbindungname}
```

Der Netzwerkanalyzer erfasste die folgenden Datenpakete während des Verbindungsaufbaus:

Number	DeltaTime	Source	Destination	Interpretation
# 1.Phase: Main Mode				
1	0 min	160.85.131.61	160.85.131.60	UDP D=500 S=500 Len=312
2	16.4 ms	160.85.131.60	160.85.131.61	UDP D=500 S=500 Len=88
3	34.6 ms	160.85.131.61	160.85.131.60	UDP D=500 S=500 Len=188
4	69.4 ms	160.85.131.60	160.85.131.61	UDP D=500 S=500 Len=188
5	**69.6 ms**	**160.85.131.61**	**160.85.131.60**	**UDP D=500 S=500 Len=1060**
6	**1.4 sec**	**160.85.131.60**	**160.85.131.61**	**UDP D=500 S=500 Len=1052**
# 2.Phase: Quick Mode				
7	3.3 sec	160.85.131.61	160.85.131.60	UDP D=500 S=500 Len=324
8	836.0 ms	160.85.131.60	160.85.131.61	UDP D=500 S=500 Len=300
9	2.6 sec	160.85.131.61	160.85.131.60	UDP D=500 S=500 Len=60

Das Diagramm protokolliert den zeitlichen Ablauf des Verbindungaufbaus IKE [→Automatische Schlüsselverbindung]. Der Unterschied zur Authentisierung mit Shared Secrets ist in der fünften und sechsten Sequenz durch die Länge der Pakete ersichtlich. Hier wird zusätzlich zum Hashwert das Zertifikat ausgetauscht.



9 Remote Access

9.1 Remote Access über ISP

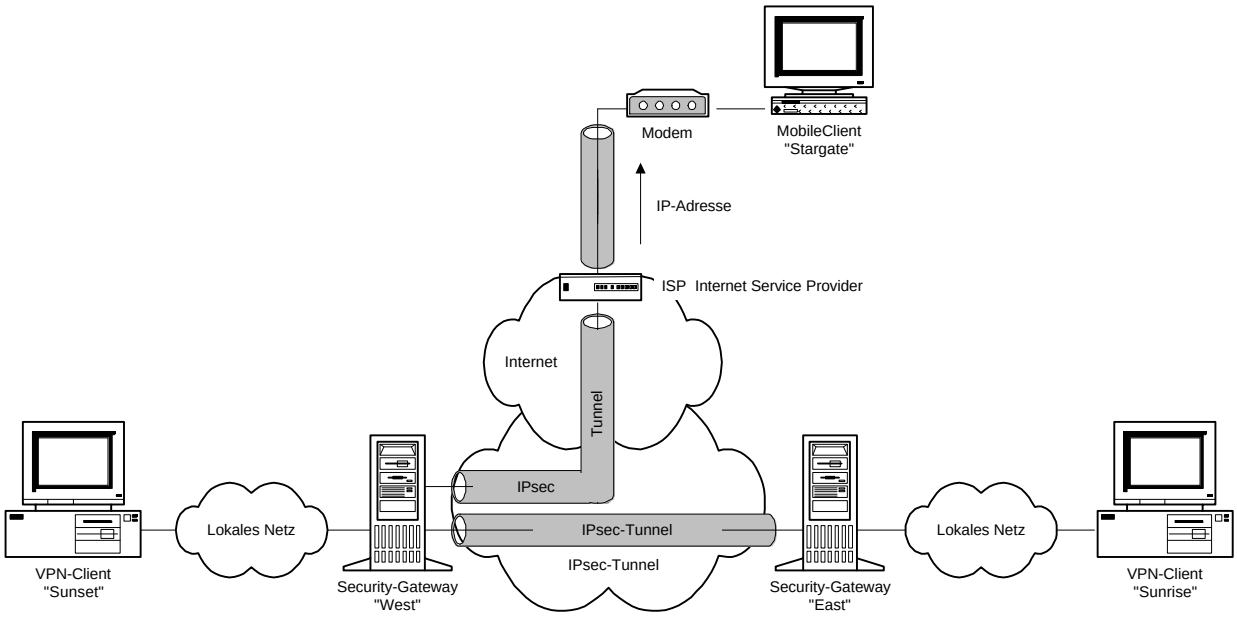


Abb. 9.1

Oft sind Netzteilnehmer über einen Internet Service Provider (ISP) ans Internet gebunden. Dazu wird mit einem Analog- oder ISDN-Modem eine Punkt-zu-Punkt-Verbindung zum ISP aufgebaut. Der Zugang über einen nahegelegenen Internet-Provider ist meist um einiges günstiger als die direkte Einwahl ins Firmennetz. Der ISP ermöglicht dem Client den Zugang zum Internet und übergibt ihm bei jedem Dial-in eine neue IP-Adresse.
Da der ISP nicht von jedermann beliebig konfiguriert werden kann, muss der Client (in unserem Fall "Stargate") als Security-Gateway eingerichtet werden, damit ein IPSec-Tunnel über das Internet geführt werden kann. Das ist aber nicht ganz unproblematisch, weil die dynamisch zugewiesene IP-Adresse des Clients zur Authentisierung mitverwendet wird. Es gibt ferner auch Lösungen der Telecom-Gesellschaften, die selbst VPN-Dienste an ihren Einwahlknoten anbieten. Dadurch entsteht das eigentliche Tunneling beim ISP [→ VPN-Architektur].
Im Zusammenhang mit IPSec werden mobile Clients auch Road Warrior genannt.

9.2 Konfiguration eines mobilen VPN-Clients

Damit überhaupt eine Verbindung zwischen einem mobilen Rechner und einem Security-Gateway aufgebaut werden kann, muss auf beiden Seiten FreeS/WAN installiert sein.
Die folgende Anleitung beschreibt die Konfiguration des mobilen VPN-Anwenders und dem gewünschten Security-Gateway. Diese Verbindungsart unterliegt leider noch einer Sicherheitslücke. Grund dafür ist die unbekannte IP-Adresse des VPN-Clients. Mit einem Trick in der Konfiguration wird diese Informationslücke überbrückt. Die Authentisierung kann also nicht mehr auf der Basis einer im Voraus bekannten IP-Adresse, sondern vielmehr auf dem Vertrauen in Zertifikaten und 'Shared Secrets' erfolgen. Die Implementation der FreeS/WAN-Software ist im derzeitigen Stadium noch nicht für diese Anwendung optimiert.

Weil dem Security-Gateway die IP-Adresse des VPN-Clients zu Beginn des Verbindungsaufbaus noch nicht bekannt ist, wird in den IPSec-Konfigurationsdateien für jede unbekannte Adresse einfach 0.0.0.0 eingetragen:

Bsp: *ipsec.conf*

```
conn mobil                          # Name der Verbindung
     left=160.85.131.60             # IP-Adresse von Gateway 'West'
     leftsubnet=10.0.1.0/24         # Lokales Netzwerk hinter 'West'
     right=0.0.0.0                  # Unbekannte 'IP-Adresse' des mobilen Clients
```

Bsp: *ipsec.secrets*

```
160.85.131.60  0.0.0.0
"0x84bd56dd_767334c8_3bad096d_a11296c2_8593b067_c6a1ba63_edb9ac1a_fa622e8f"
```

Bei einer Authentisierung mittels 'Shared Secrets' bedeutet das, dass alle Clients die gleichen Schlüssel teilen, was aus sicherheitstechnischem Blickwinkel nicht als optimale Methode angesehen werden kann. Diese Sicherheitslücke wird jedoch mit der Einführung von Zertifikaten weitgehend geschlossen. Als weiterer Nachteil muss erwähnt werden, dass FreeS/WAN nur eine Verbindung zu einem mobilen Client gleichzeitig aufbauen kann.

Falls die Verbindung über einen ISP aufgebaut wird, funktioniert die Konfiguration des mobilen Clients wie in der in [Startup-Script] beschriebenen Prozedur.

Für jede neue VPN-Verbindung muss ein Namen assoziiert werden. In unserem Fall heisst die Verbindung 'mobil'. In der Datei *ipsec.conf* werden dann alle Verbindungen spezifisch konfiguriert. Mit der Include-Anweisung hat man die Möglichkeit auf eine weitere Datei zu verweisen. Dadurch ist es möglich für jede VPN-Verbindung eine separate Konfigurationsdatei anzulegen.
Beispiel: Der folgende Eintrag in *etc/ipsec.conf* verweist auf eine weitere Konfigurationsdatei.

```
include ipsec.2.conf  # Conf-Datei für Remote-Access-Konfiguration
```

Der Verbindungsaufbau geschieht wie in [Automatische Schlüsselverbindung] beschrieben, er muss aber auf jeden Fall vom mobilen Client initiiert werden.

9.3 Startup-Script

Durch die Modemverbindung zum ISP wird dem mobilen Rechner eine dynamische IP-Adresse zugewiesen. Das erschwert den Vorgang für einen Verbindungsaufbau via IPSec, weil die erhaltene IP-Adresse und diejenige des ISP's zuerst mit dem Befehl *ifconfig* abgefragt werden, und manuell in die Konfigurationsdatei eingetragen werden müssen. Zudem besteht das Problem, dass das Netzwerk-Device des Modems ppp0 erst als solches erkannt wird, wenn man sich beim ISP eingewählt hat. Der Pluto-Daemon ändert beim Aufstarten von Linux automatisch die Route-Einträge auf das virtuelle ipsec0 Device. Das funktioniert aber nicht fehlerfrei, weil das Device ppp0 zu diesem Zeitpunkt noch nicht zur Verfügung steht. Darum sind folgende Schritte in der exakten Reihenfolge notwendig um einen Verbindung via IPSec aufzubauen:

`wvdial` Mit diesem Befehl wird eineVerbindung zum ISP gemacht.

In neuem Terminalfenster:

```
cd /usr/local/lib/ipsec
./ipsec setup –restart
```

`ifconfig ppp0` Dyn. IP-Adresse des Clients (inet addr) und des ISP (P-t-P) herauslesen und in die Konfigurationsdatei ipsec.conf unter *right* und *rightnexthop* eintragen. Datei speichern.

```
./ipsec auto –show –add {Verbindungsname}
./ipsec auto –show –up  {Verbindungsname}
```

Dieser komplizierte Vorgang ist für Anwender nicht zumutbar. Darum haben wir ein Script geschrieben, welches die obengenannten Vorgänge weitgehend selbständig ausführt. Die IP-Adressen werden eingelesen, an Umgebungsvariablen übergeben und in die Konfigurationsdatei eingetragen. Die Script-Datei hat den Name ***startup*** und kann in ein Verzeichnis wie */bin* oder */usr/local/bin* kopiert werden. In der Konfigurationsdatei *ipsec.conf* müssen nun, anstatt jedesmal die IP-Adressen anzupassen, einmal die folgenden Einträge gemacht werden:

```
right=$DYNIP
rightnexthop=$PEERIP
```

Nun geschieht der Verbindungsaufbau wie folgt:

```
wvdial
startup {Verbindungsname}
```

Das Listing des Scripts ist im Anhang unter [Startup-Script] aufgeführt.

9.4 Virtuelles Subnetz

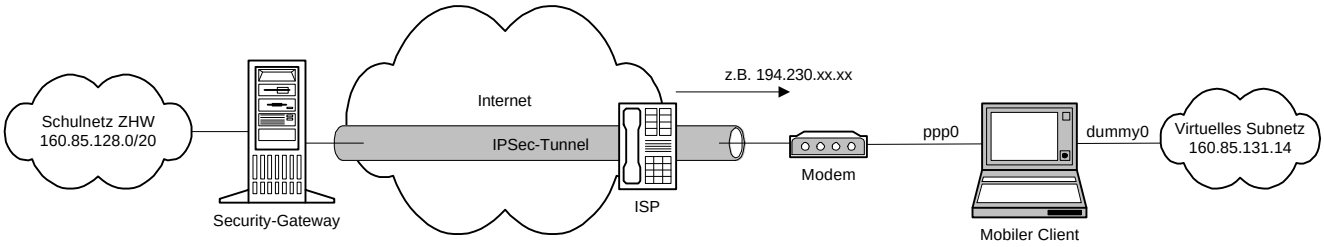


Abb. 9.2

Das Modell der Abbildung 9.2 zeigt, wie ein mobiler Client mit einer dynamischen IP-Adresse über einen VPN-Tunnel mit einem anderen Netzwerk (z.B. Schulnetz) verbunden ist. Anstoss für die Überlegung ist die Idee, mobile Rechner über ein VPN ans Schulnetz der ZHW zu binden. So könnten Mitarbeiter und Studenten von zu Hause aus eine sichere Verbindung zum Schulnetz aufbauen. Diese Überlegung bringt aber zugleich gewisse Nachteile mit sich. Mit seiner vom ISP erhaltenen Internet-Adresse hat der Benutzer im Schulnetz praktisch keine Rechte, weil er nicht als offizieller Schulrechner registriert ist. Abhilfe dafür könnte man schaffen, indem man den VPN-Clients einen gewissen Adressbereich des Schulnetzes (160.85.128.0/20) reserviert. Dadurch unterscheidet sich ein mobiler Client nicht mehr von einem normalen Schulrechner.
Die Idee ist gut, aber nicht so einfach zu realisieren. Die Schwierigkeit liegt an der Konfiguration des VPN-Clients. Er soll zwar mit der dynamischen IP-Adresse vom ISP einen Tunnel zum Security-Gateway der Schule bilden, trotzdem aber eine IP-Adresse des Schulnetzes besitzen. Aus diesem Grund wurde ein sogenanntes virtuelles Subnetz beim VPN-Client eingerichtet. Das geschieht durch Konfiguration eines Dummy-Netzwerk-Devices, das in Wirklichkeit gar nicht existiert. Diesem kann eine lokale IP-Adresse zugeteilt werden. Dazu muss in der Konfigurationsdatei *etc/rc.config* der folgende Eintrag stehen.

```
SETUPDUMMYDEV='yes'
```

In der Netzwerkkonfiguration kann das virtuelle Netzwerk-Device eingeführt werden (IP-Adresse ist fiktiv):

Nummer	Aktiv	Netzwerktyp	Device-Name	IP-Adresse
[0]	**[X]**	**Unbekannt**	**dummy0**	**160.85.131.14**

Nun muss IPSec für das neue Subnetz konfiguriert werden. Der folgende Eintrag in der Datei *ipsec.conf* muss bei beiden Kommunikationspartner vorhanden sein:

```
rightsubnet=160.85.131.14/32
```

9.4.1 Ungelöstes Problem mit dynamischer IP-Adresse

Mit der Realisierung dieser Idee kann aber nur das halbe Problem gelöst werden. Die Kommunikation vom Schulnetz zum mobilen Rechner über den VPN-Tunnel funktioniert auf diese Weise ausgezeichnet. Leider hat aber der VPN-Client keinen Zugang mehr. Wo liegt das Problem?
Voraussetzung für das Tunneling eines Paketes ist, dass es die richtige Source- und Destination- Adresse hat. Genau dies ist im Fall des mobilen Client mit seinem virtuellen Subnetz nicht gegeben.
Damit das Paket getunnelt wird, muss es als Source-Adresse die IP-Adresse des Dummy-Devices haben, nämlich 160.85.131.14 und als Destination Adresse 160.85.128.x . Genau diese Voraussetzung ist nicht gegeben. Damit dieses Problem genauer analysiert werden kann, muss man die Routing Tabelle des mobilen Clients näher anschauen.

```
bash-2.03# route

Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
pop-zh-7.freesu *               255.255.255.255 UH    0      0        0 ppp0
pop-zh-9-2.free *               255.255.255.255 UH    0      0        0 ipsec0
160.85.128.0    pop-zh-9-2.free 255.255.255.0   UG    0      0        0 ipsec0
160.85.131.14   *               255.255.255.0   U     0      0        0 dummy0
loopback        *               255.0.0.0       U     0      0        0 lo
default         pop-zh-7.freesu 0.0.0.0         UG    0      0        0 ppp0
```

Es gilt:
pop-zh-7.freesu (DNS des Standard Gateways des ISP) → 194.230.3.175
pop-zh-9-2.free (DNS der dynamischen Adresse vom ISP) → 194.230.193.20

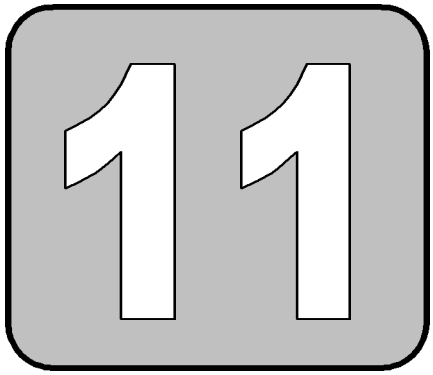
Wie man sieht, werden Datenpakete mit der Destination Adresse 160.85.128.x zur Netzwerkkarte, welche die dynamische IP-Adresse besitzt, geschickt. Dadurch besitzen alle Datenpakete mit dieser Destination-Adresse als IP-Source Adresse 194.230.193.20 . Aus diesem Grund werden die Datenpakete, obwohl sie am richtigen Interface rausgehen, nicht getunnelt.
Damit aber das Tunneling funktioniert, muss es über das Dummy-Device zum IPSec-Interface kommen. Dadurch würde es die richtige Source-Adresse bekommen und somit getunnelt werden.
Folgende Versuche wurden gestartet, um dies zu erreichen:

- Verschiedene Einstellungen in der Routing Tabelle, so dass die Datenpakete mit dem Umweg über das Dummy-Device zum Ipsec-Interface kommen.
- Mittels Ipchain konfiguration (Firewall-Konfiguration)

Parallel zu den Versuchen wurde das Mailing-Archiv nach Lösungen für unser Problem durchsucht. Aber weder die Suche in der Mailingliste noch unsere zahlreichen Versuche haben zum Erfolg geführt.

10 Zeitplan

Ausgeführte Tätigkeiten	Zeitplan vom 9.Sept 99 - 1.Nov 99																																					
	KW36		KW37					KW38					KW39					KW40					KW41					KW42					KW43					
Einarbeitung in Thematik	X	X	X				X																															
Installation Linux		X																																				
Netzwerk-Konfiguration			X	X	X																																	
Installation vov FreeS/WAN						X	X																															
Inbetriebnahme von IPSec							X	X																														
Paket-Analyse										X	X																											
IKE-Analyse											X	X	X	X	X																							
Simulation Replay-Angriff												X	X																									
Konfiguration Road Warrior													X	X	X																							
Einarbeitung Zertifikate																X	X	X																				
Installation OpenSSL / Patch																	X	X																				
IKE mit Zertifikaten																		X	X	X	X		X															
Remote Access																									X	X	X											
Versuche mit virtuellem Subnetz																											X	X	X	X	X	X						
Dokumentation									X				X	X						X	X	X		X	X	X	X					X	X	X	X	X	X	
Binden / Abgabe																																						X



11 Ausblick

11.1 Wie weiter?

Die geleistete Arbeit bietet eine gute Basis für weitere Arbeiten. Das von uns gewonnene Wissen kann anhand dieser Dokumentation weiterverwendet werden. Es ist längerfristig durchaus realistisch ein VPN mit IPSec-Technologie in einem professionellen Umfeld einzusetzen. Man muss sich aber bewusst sein, dass IPSec noch mitten in der Entwicklungsphase steckt.

11.2 Neue Version IPSec FreeS/WAN

In der zweitletzten Woche unserer Arbeit wurde die FreeS/WAN Version 1.1 publiziert. In dieser Version wurden vor allem kleinere Bugs behoben. FreeS/WAN sollte nun voll kompatibel mit Kernel 2.0.38 und auch 2.2.12 oder höher sein. Zur gleichen Zeit erschien auch eine erweiterte Version vom Pluto-Patch. Dieser soll nun auch die Zertifikatsverwaltung über eine Berkeley-Datenbank oder LDAP unterstützen.
Wir haben versucht die neuen Versionen zu installieren, sind aber am Pluto-Patch gescheitert. Nach Behebung einiger überflüssigen Einträge in der Patch-Datei selber, müssen Verzeichnisverweise in der Datei `/usr/src/freeswan-snap1999MonthDayb/pluto/Makefile` angepasst werden. Weiter verlangt der Pluto-Patch, dass die Programme OpenSSL, OpenLDAP und Kerberos 5.1 installiert sind, damit der Kernel erfolgreich kompiliert werden kann. Kerberos ist ein vom MIT entwickelter Authentifizierungsmechanismus, der auf symmetrischen, kryptographischen Verschlüsselungsverfahren basiert. Leider konnte dieses Programm nicht innerhalb nützlicher Frist besorgt werden, weil es nicht für Export aus der USA bestimmt ist. Das hat uns einmal mehr demonstriert, wie mühsam und zeitraubend solche Installationen sein können, und mussten hier deshalb aus Zeitgründen abbrechen.

11.2.1 Zertifikatsverteilung über LDAP

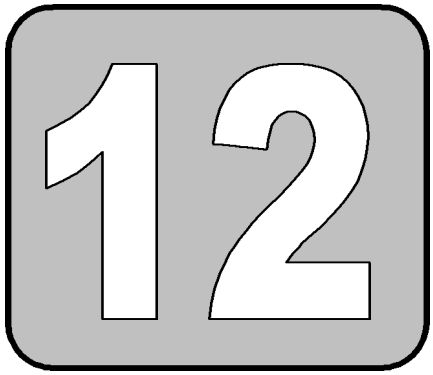
Ein weiteres Ziel das zu verfolgen wäre, ist die Zertifikatsverteilung über *LDAP*. Die Zertifikate könnten somit von einer CA mit *LDAP*-Server zentral verwaltet werden. Dies ist in der neusten Version vom Pluto-Patch implementiert, konnte aber aus Zeitgründen nicht mehr realisiert werden.

11.2.2 Graphische Benutzeroberfläche

Die Bedienerfreundlichkeit ist ein weiterer Schwachpunkt der Linux Entwicklungen. Um ein Produkt auch kommerziell Vermarkten zu können, muss eine einfache Handhabung gewährleistet sein. Als Fortsetzung unserer Arbeit könnte eine graphische Benutzeroberfläche (GUI) programmiert werden, welche die Konfigurierung des VPN's erleichtert.

11.2.3 Testphase

Als nächster Schritt wäre eine intensive Testphase nötig. In dieser würde zum einen die Stabilität und Zuverlässigkeit der Chiffrierung getestet, zum anderen die Performance auf Grund vom Datendurchsatz und Bandbreite ermittelt. Anhand den Testergebnissen kann entschieden werden, in welchen Gebieten IPSec bereits eingesetzt werden kann. Da wir diesen Schritt nicht mehr machen konnten, wollten wir dem Leser doch eine Vorstellung über die Leistungsfähigkeit des FreeS/WAN Systems nicht Vorenthalten, aus diesem Grund haben wir im Archiv der 'Mailing List' eine Angabe [→ Anhang 14.6] gesucht und gefunden.



12 Schlusswort

12.1 Schlusswort

Diese Diplomarbeit war ein Glücksfall für uns. Wir bekamen genau die Arbeit, die uns am meisten interessierte. Die Thematik Internetsicherheit faszinierte uns während der ganzen Diplomarbeit, nicht zuletzt, weil es heute und in Zukunft ein wichtiges Thema bleiben wird.

Informationsbeschaffung über das Thema ‘Internet-Sicherheit‘ und im speziellen ‘Virtual Private Networks‘ und dessen Komponenten war kein Problem. Da die Nachfrage für dieses Produkt gross ist, gibt es zahlreiche Bücher darüber und auch die Computer Zeitschriften geizen nicht mit Beiträgen über dieses Thema.

Um unseren Arbeitsplatz und unsere Hardware Komponenten wurden wir äusserst beneidet. Wir hatten einen grossen Raum für uns allein zur Verfügung. Dies war auch nötig, denn parallel zur praktischen Arbeit gab es immer wieder Momente, wo eine intensive Auseinandersetzung mit der doch sehr anspruchsvollen Theorie gefragt war. Dabei kam uns eine ruhige Atmosphäre sehr entgegen. Die vier leistungsfähigen Computer mit den vier Flachbildschirmen blieben unseren Kollegen nicht verborgen, wir haben viele neidvolle Sprüche und Blicke ertragen müssen, aber zugleich auch genossen.

Die Zusammenarbeit mit Herrn Steffen hat sehr gut geklappt. Er hat uns mit seinen abgegebenen Informationen am Anfang einen guten Einstieg in diese Diplomarbeit gegeben. Bei Fragen und Problemen war er stets per E-Mail oder per Natel erreichbar. Zwischen unseren Fragen und seinen Antworten gab es keine zeitlichen Verzögerungen. Besonders bei den Zertifikaten konnte wir von seinem Vorwissen profitieren, so dass wir unsere Anfangsschwierigkeiten mit dieser Thematik relativ schnell in den Griff bekamen.

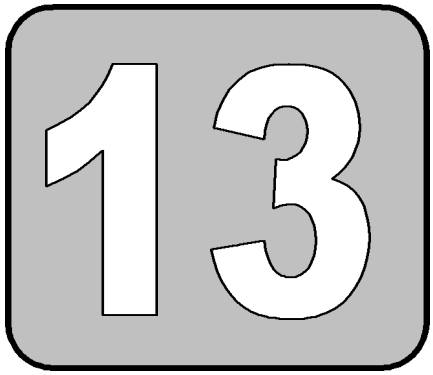
Eine weitere sehr hilfreiche Quelle war die FreeS/WAN-Mailing-List. Obwohl nicht jede unserer Fragen beantwortet worden ist, konnten wir doch von den Fragen und Antworten anderer einiges dazulernen.

Trotzdem wollen wir erwähnt haben, dass im Grunde genommen eine ‘halbe‘ Projektarbeit erforderlich ist, nur um das Linux einigermassen zu verstehen. Einige Male hatten wir unsere Problem mit Linux und immer mussten wir unsere Unerfahrenheit in Form von Zeitverlusten bezahlen. Aus diesem Grund hätten wir es sicher sehr begrüsst, wenn wir zuerst eine Projektarbeit als Vorarbeit auf diese Diplomarbeit hätten machen können, denn die Materie Internet-Sicherheit war im Grunde genommen schwierig genug.

Als ein weiteres Manko empfanden wir zum einen die spärlichen Information zu den verschiedenen VPN-Anwendungen, und zum anderen die schwer verständliche Installationsanweisung zum OpenSSL Patch. Diese zu installieren war eine äusserst mühsame Angelegenheit. Uns ist es wohl bewusst geworden, gerade im Rahmen dieser Arbeit, wie schwer es ist, eine gute Installationsanweisung zu schreiben. Trotzdem hätten wir doch in dieser Hinsicht mehr von diesen ‘Linux-Gurus‘ erwartet.

12.2 Danksagung

Wir möchten uns bei allen Personen, die uns beim Gelingen unserer Arbeit unterstützt haben bedanken. Besonderen Dank verdient Herr Steffen, der uns nie im Stich liess, wenn wir mal in eine VPN-Sackgasse gerieten. Dank geht auch an Roger Zuber, der uns immer anstandslos mit neuer Hardware belieferte; sowie dem IPSec-Entwickler-Team, die uns über die Mailingliste immer wieder mit neuen Ideen und Lösungen beglückten.
Und last but not least, an unsere Mitstudenten, die immer wieder gut für einen ablenkenden Schwatz waren und für gute Stimmung sorgten, so dass diese Zeit uns noch lange in guter Erinnerung bleiben wird.



13 Quellen

Links

[1]	Sämtliche RFCs:	http://www.ietf.org
[2]	FreeS/WAN Projekt:	http://www.xs4all.nl/~freeswan
[3]	FreeS/WAN OpenSSL Patch:	ftp://hplose.hpl.hp.com/pub/nd/pluto-openssl.tar.gz
[4]	OpenSSL Projekt:	http://www.openssl.org
[5]	X.509 Zertifikate:	http://freecerts.entrust.com
[6]	FreeS/WAN Mailing List:	mailto:linux-ipsec@clinet.fi
[7]	FreeS/WAN Mailing Archiv	http://www.sandelman.ottawa.on.ca/linux-ipsec/

Literatur

[8] Dave Kosiur, ”Virtual Private Networks”, John Wiley & Sons, Toronto, 1998, ISBN 0-471-29526-4

[9] Richard E. Smith, “Internet-Kryptographie”, ADDISON-WESLEY, Bonn, 1998 ISBN 3-8273-1344-9

[10] Charly Scott, Paul Wolfe & Mike Erwin, “Virtuelle Private Netzwerke”, O’REILLY, Köln, 1999, ISBN 3-89721-123-8

[11] Kai Fuhrberg, “Internet-Sicherheit”, Hanser, München, 1998, ISBN 3-446-19400-2

[12] Christian Reiser, “Internet-die Sicherheitsfragen”, Ueberreuter, Wien, 1998, ISBN 3-7064-0420-6

[13] Othmar Kyas, “Sicherheit im Internet”, Datacom, Bonn, 1998, 2.Aufl. , ISBN 3-8266-4024-1

[14] Andrew S. Tannenbaum, “Computer-Netzwerke”, Prentice Hall, München, 1998, ISBN 3-8272-9568-8

[15] Frank Bitzer, Klaus M. Brisch, “Digitale Signatur”, Springer, Berlin, 1999, ISBN 3-540-65563-8

Redbooks [www.redbooks.ibm.com]

[16] SG24-5404-00 AS/400 Internet Security: Implementing AS/400 VPNs, Kapitel 1
[17] SG24-5201-00 Comprehensive Guide to Virtual Private Networks, Kapitel 2 ,3, 5
[17] SG24-5209-00 Comprehensive Guide to Virtual Private Networks, Vol. 3, Kapitel 4
[18] GG24-3376-05 TCP/IP Tutorial and Technical Overview, Kapitel 2 und 5

Scripts

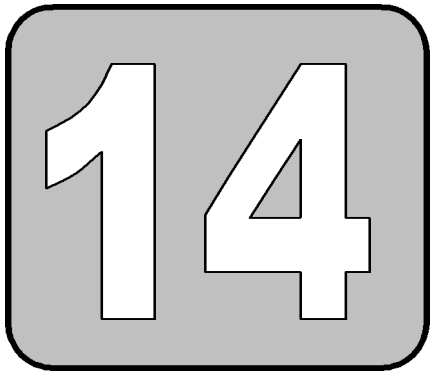
[19] Manfred Lindner, “Datacommunication and Networks”, Cisco SE1 Starter Course, 1998
[20] Christian Ewald, Roger Orell, “Intranet complet”, FHW Fach GKT Projektarbeit 1, 1999

Artikel von Fachzeitschriften

[21] Michael Schmidt, “Schwan und Pinguin”, c’t 16/98, S 180-185
[22] Michael Schmidt, “Unter Ausschluss der Öffentlichkeit”, c’t 8/98, S 226-235
[23] Jürgen Kuri, “Privatissimo”, c’t 4/99, S 190-194
[24] Kai Martius, “Paketschutz”, iX 6/98, S 124-129
[25] Kai Martius, “Schlüsselwerkzeug”, iX 8/98, S 107-111
[26] Martin Raepple, “Transportsicherung”, iX 1/99, S 118-122
[27] Keine Angabe, “Deine Angst vor Schnüfflern”, Gateway 12/98
[28] Keine Angabe, “Safer Net”, Gateway 5/99
[29] Keine Angabe, “Geschlossene Gesellschaft”, Gateway 4/99
[30] Keine Angabe, “Sicherheit von der Stange”, Gateway 8/99
[31] SuSE, “Installation, Konfiguration und erste Schritte mit SuSE Linux 6.2 ”, 15. Auflage 99

RFC Request for Comment [www.zhwin.ch/~sna/research/VPN/]

[32] RFC 2409 The Internet Key Exchange (IKE)
[33] RFC 2408 Internet Security Association and Key Management Protocol (ISAKMP)
[34] RFC 2407 The Internet IP Security Domain of Interpretation
[35] RFC 2402 IP Authentication Header (AH)
[36] RFC 2406 IP Encapsulating Security Payload (ESP)



14 Anhang

Im Anhang befinden sich Listings, Konfigurationsbeispiele, Zertifikate,Testresultate, etc...

14.1 Listing der Script-Datei startup

```
#!/bin/bash
#Script zum automatischem IPSec-Verbindungsaufbau über Modem
if test $# -ne 1 ; then
        echo "Falsche Eingabe: startup {Verbindungsname}"
        exit 1
fi

cd /usr/local/lib/ipsec
./ipsec auto --show --delete $1
./ipsec setup --restart
echo "Hole IP-Adresse"
export DYNIP=`ifconfig ppp0 | grep inet | awk '{print $2}' | \
sed 's/^addr://g'`
export PEERIP=`ifconfig ppp0 | grep inet | awk '{print $3}' | \
sed 's/^P-t-P://g'`
readip
cd /usr/local/lib/ipsec
./ipsec auto --show --add $1
echo "OK"
echo "Verbindungsaufbau"
./ipsec auto --show --up $1
```

14.2 Inhalt eines Zertifikats

```
issuer :/C=CH/ST=Winterthur\x08\x08/L=Winterthur/O=Tech/Email=e5uenal@zhwin.ch
subject:/C=CH/ST=Some-State/O=Tech/CN=ksy008.zhwin.ch IPsec certificate
#1/Email=e5gaertn@zhwin.ch
serial :07

Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number: 7 (0x7)
        Signature Algorithm: md5WithRSAEncryption
        Issuer: C=CH, ST=Winterthur\x08\x08, L=Winterthur, O=Tech/Email=e5uenal@zhwin.ch
        Validity
            Not Before: Oct 11 13:54:03 1999 GMT
            Not After : Oct 10 13:54:03 2000 GMT
        Subject: C=CH, ST=Some-State, O=Tech, CN=ksy008.zhwin.ch IPsec certificate
#1/Email=e5gaertn@zhwin.ch
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
            RSA Public Key: (1024 bit)
                Modulus (1024 bit):
                    00:dd:1b:19:18:c6:37:31:28:6a:72:9f:cc:e9:e0:
                    d5:8f:ff:ec:fc:eb:06:29:6b:c3:2b:be:51:95:b0:
                    3c:c7:f9:2f:98:fb:8d:0d:bd:a1:13:73:70:64:ac:
                    4e:ae:e2:94:c4:80:ac:f9:8c:70:28:91:77:88:e2:
                    11:03:1e:a3:78:fb:76:db:c8:d6:0a:9c:89:4b:9e:
                    2a:eb:c1:62:90:9f:30:9c:0b:30:7e:e4:c7:1e:7c:
                    a4:3d:b3:f5:ec:db:cc:4e:97:4c:f5:32:1e:3e:b4:
                    2d:07:0a:6f:fb:13:78:43:97:60:25:98:7f:c2:36:
                    76:05:67:3a:11:98:49:86:d3
                Exponent: 65537 (0x10001)
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
            Netscape Cert Type:
                SSL Server
            X509v3 Key Usage:
                Digital Signature, Non Repudiation, Key Encipherment
            Netscape Comment:
                IPsec certificate for ksy008.zhwin.ch
            X509v3 Subject Key Identifier:
                89:18:79:E9:9F:E9:81:9B:E7:B3:9F:67:80:82:BE:F8:E4:6B:00:DB
            X509v3 Authority Key Identifier:
                keyid:5A:5D:9E:FE:A9:A5:57:4A:99:AD:EE:5D:E3:0B:6A:EA:8F:8A:03:24
                DirName:/C=CH/ST=Winterthur\x08\x08/L=Winterthur/O=Tech/Email=e5uenal@zhwin.ch
                serial:00

            X509v3 Subject Alternative Name:
                email:e5gaertn@zhwin.ch, DNS:ksy008.zhwin.ch, IP Address:160.85.131.60
            X509v3 Issuer Alternative Name:
                email:e5uenal@zhwin.ch
    Signature Algorithm: md5WithRSAEncryption
        21:6a:ff:c4:06:4e:de:d2:82:67:41:24:d3:68:b8:97:c4:ad:
        0f:dd:7d:c4:1f:3f:dd:f3:94:41:04:7d:77:d2:30:49:e9:8f:
        3c:97:af:1e:63:71:8d:45:e7:17:56:f3:db:99:9b:92:e1:68:
        7a:ac:f8:fe:1d:04:89:57:f2:e1:9b:b0:cb:a7:38:6f:0b:3a:
        ad:21:c8:41:71:94:6b:2e:47:5e:10:4c:ad:0e:f0:16:d0:ce:
        07:34:15:db:38:61:93:42:7c:1a:35:59:52:e7:62:01:e4:c0:
        b1:f4:f7:1b:f1:90:e9:c4:8a:5f:f6:a6:1e:b4:d9:66:c0:ae:
        00:57

-----BEGIN CERTIFICATE-----
MIID0DCCAzmgAwIBAgIBBzANBgkqhkiG9w0BAQQFADBpMQswCQYDVQQGEwJDSDEV
MBMGA1UECBYMV2ludGVydGh1cggIMRMwEQYDVQQHEwpXaW50ZXJ0aHVyMQ0wCwYD
VQQKEwRUZWNoMR8wHQYJKoZIhvcNAQkBFhBlNXVlbmFsQHpod2luLmNoMB4XDTk5
MTAxMTEzNTQwM1oXDTAwMTAxMDEzNTQwM1owgYIxCzAJBgNVBAYTAkNIMRMwEQYD
VQQIEwpTb21lLVN0YXRlMQ0wCwYDVQQKEwRUZWNoMS0wKwYDVQQDFCRrc3kwMDgu
emh3aW4uY2ggSVBzZWMgY2VydGlmaWNhdGUgIzExIDAeBgkqhkiG9w0BCQEWEWU1
Z2FlcnRuQHpod2luLmNoMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDdGxkY
xjcxKGpyn8zp4NWP/+z86wYpa8MrvlGVsDzH+S+Y+40NvaETc3BkrE6u4pTEgKz5
jHAokXeI4hEDHqN4+3bbyNYKnIlLnirrwWKQnzCcCzB+5McefKQ9s/Xs28xOl0z1
Mh4+tC0HCm/7E3hDl2AlmH/CNnYFZzoRmEmG0wIDAQABo4IBbDCCAWgwCQYDVR0T
BAIwADARBglghkgBhvhCAQEEBAMCBkAwCwYDVR0PBAQDAgXgMDQGCWCGSAGG+EIB
DQQnFiVJUHNlYyBjZXJ0aWZpY2F0ZSBmb3Iga3N5MDA4Lnpod2luLmNoMB0GA1Ud
DgQWBBSJGHnpn+mBm+ezn2eAgr745GsA2zCBkwYDVR0jBIGLMIGIgBRaXZ7+qaVX
Spmt7l3jC2rqj4oDJKFtpGswaTELMAkGA1UEBhMCQ0gxFTATBgNVBAgWDFdpbnRl
cnRodXIICDETMBEGA1UEBxMKV2ludGVydGh1cjENMAsGA1UEChMEVGVjaDEfMB0G
CSqGSIb3DQEJARYQZTV1ZW5hbEB6aHdpbi5jaIIBADAzBgNVHREELDAqgRFlNWdh
```

```
ZXJ0bkB6aHdpbi5jaIIPa3N5MDA4Lnpod2luLmNohwSgVYM8MBsGA1UdEgQUMBKB
EGU1dWVuYWxAemh3aW4uY2gwDQYJKoZIhvcNAQEEBQADgYEAIWr/xAZO3tKCZ0Ek
02i4l8StD919xB8/3fOUQQR9d9IwSemPPJevHmNxjUXnF1bz25mbkuFoeqz4/h0E
iVfy4Zuwy6c4bws6rSHIQXGUay5HXhBMrQ7wFtDOBzQV2zhhk0J8GjVZUudiAeTA
sfT3G/GQ6cSKX/amHrTZZsCuAFc=
-----END CERTIFICATE-----
```

14.3 Inhallt eines privaten Schlüssels

```
-----BEGIN RSA PRIVATE KEY-----
MIICXAIBAAKBgQDdGxkYxjcxKGpyn8zp4NWP/+z86wYpa8MrvlGVsDzH+S+Y+40N
vaETc3BkrE6u4pTEgKz5jHAokXeI4hEDHqN4+3bbyNYKnIlLnirrwWKQnzCcCzB+
5McefKQ9s/Xs28xOl0z1Mh4+tC0HCm/7E3hDl2AlmH/CNnYFZzoRmEmG0wIDAQAB
AoGAVsUGoVFQYYtB0v/hU7s3NE60O0EkZubybd0/eSXzhGrHAhtd7JGJf++ioO95
5WnoBgGVYE17yn/j0CUflgDVF6+FpFfwMi204UOvECs6786gHWOQYeU4naxXwH5n
U4yxuy4YFfxFZgmotfgdP3du74kN7On3EVLLocBp6zA6DukCQQD2fi4E3q4sc7Be
fgijGBsI8ZrH9ppd9QUA3sw86hqvdVm0MrRih1K5P9Kyc+Ne0W9Q3YtNezI2hRaj
TRyS84ZFAkEA5aJAeksXOHZd8CbXfUxUi3yl80UlUcWtp7Z26H2aQMCB1/i8+F28
Ct1xA2+Cob7bmfpBvujHXSDf1ZOtFXLWNwJBAKiP90ZrRW6umpCz3ZIyvOIr3q4a
LMxN72L9+Ws6qI8OUA7TkdnDMsXuKFbSu00vKdGB/2437kmsT6PS6tRvZoUCQFl5
nyYHnESj2nZ4q/5mf4raRs9DYgc6vy9aKXrtde1FFVLR2M/ttheFsUNlb8EWXPri
SDva99ORu6W8pPR2ioECQEDigdzU55IIbTccfpkhYKfhXhqto8n2KuMlADBoZlWS
YDpqs5C/Viv3y+ekit5WWebNNX4WVpimS6vbh8SkNts=
-----END RSA PRIVATE KEY-----
```

14.4 Beispiel Konfigurationsdatei ipsec.conf

```
# /etc/ipsec.conf - FreeS/WAN IPSEC configuration file

# CAUTION
# If you are using manually-keyed connections for more than just preliminary
# testing, encryption/authentication keys for those connections should be
# put in a separate file (with permissions rw-------) using the also parameter
# and the include facility -- see the ipsec.conf(5) manpage -- so that the
# keys are not generally readable.

# basic configuration
config setup
        # virtual and physical interfaces for IPSEC, normally a single
        # `virtual=physical' pair, or a (quoted!) list of pairs.  In the
        # simple case, where you only want to run IPSEC on one interface,
        # the virtual (ipsec0) shouldn't need changing but the physical
        # (eth999) will (to the interface connecting to the public network,
        # e.g. eth0 or ppp0 or something like that).
        # *This must be right* or almost nothing will work.
        interfaces="ipsec0=eth0"
        # should setup turn IP forwarding on after IPSEC is started, and off
        # before it is stopped?
        forwardcontrol=no
        # KLIPS debugging output.  "none" for none, "all" for lots
        klipsdebug=all
        # Pluto debugging output.  "none" for none, "all" for lots
        plutodebug=all
        # manually-keyed connections to set up at startup
        manualstart=
        # connections to load into Pluto's internal database at startup
        plutoload=
        # connections for Pluto to try to negotiate at startup
        plutostart=
        # should Pluto wait for each negotiation to finish before proceeding?
        plutowait=yes

# connection specifications

# sample tunnel (manually or automatically keyed)
# "(manual)" means relevant only to manual keying, "(auto)" only to automatic.
# For manual keying, we use ESP for both encryption and authentication, the
# simplest and often the best method.
# The network here looks like:
#   leftsubnet====left----leftnexthop......rightnexthop----right====rightsubnet
# If left and right are on the same Ethernet, omit leftnexthop and rightnexthop.
conn sample
        type=tunnel
        # left security gateway (public-network address)
        left=160.85.131.60
        # next hop to reach right
        #leftnexthop=
        # subnet behind left (omit if there is no subnet)
        leftsubnet=10.0.1.0/24
        #leftfirewall=yes
        # right s.g., subnet behind it, and next hop to reach left
        right=160.85.131.61
        #rightnexthop=
        rightsubnet=10.0.2.0/24
        #rightfirewall=yes
        # (manual) base for SPI numbering; must end in 0
        spibase=0x200
        # (manual) encryption/authentication algorithm and parameters to it
        esp=3des-md5-96
        espenckey=0xf83f6c87_f84b9ae4_43764981_86544c6f_c1d4d360_af0b860e
        espauthkey=0xd37984f1_13070cf2_e0eaf4f8_01cd414b
        # (auto) key-exchange type
        keyexchange=ike
        # (auto) key lifetime (before automatic rekeying)
        keylife=2h
        #rekeymargin=5s
        # (auto) how persistent to be in (re)keying negotiations (0 means very)
        keyingtries=0
        #certopts=!send

include ipsec.2.conf  # Conf-Datei für Roadwarrior-Konfiguration
include ipsec.3.conf  # Conf-Datei für Certifcate-Authentification
include ipsec.4.conf  # Conf-Datei für Remote-Access
```

14.5 Beispiel Konfigurationsdatei ipsec.secrets

```
# This file holds shared secrets which are currently the only inter-Pluto
# authentication mechanism.  See ipsec_pluto(8) manpage.  Each secret is
# (oversimplifying slightly) for one pair of negotiating hosts.

# The shared secrets are arbitrary character strings and should be both
# long and hard to guess.  A long hex number is supplied automatically as
# an example just because it's convenient to generate.

# Note that all secrets must now be enclosed in quotes, even if they have
# no white space inside them.
#
#Secret für Site-to-Site-Verbindung#
#
160.85.131.60 160.85.131.61
"0xba078756_76cb52c6_1ca1bd29_2b775e0d_7cd2ac38_8718463f_3542049d_da1a7a72"
#
#Secret für Remote Access#
#
160.85.131.60  0.0.0.0
"0x84bd56dd_767334c8_3bad096d_a11296c2_8593b067_c6a1ba63_edb9ac1a_fa622e8f"
#
```

14.6 Performancetest FreeS/WAN

```
Tunnel mode, AH:        26.2 %
Transport mode, AH:     26.6%
CPU server:             50%
CPU client:             25%

Tunnel mode, ESP:       9.7%
Transport mode, ESP:    10%
CPU server:             55%
CPU client:             35%

Tunnel mode, AH+ESP:    7.6%
Transport mode, AH+ESP: 7.8%
CPU server:             60%
CPU client:             45%


The total throughput (100%) is 10480.94 Kbps , measured with ttcp.
The CPU usage values are rough estimations. I used top to average the %
system cpu usage, but this is not very exact because all system tasks
are considered.

Client and Server:
PII 233 Mhz, 128 Mb RAM, 100 Mbps connection (at 10480.94 Kbps), kernel
2.0.36, FreeS/Wan 1.0
```

14.7 Datenträger

Alle unsere verwendeten Dateien sind auf einem beigelegten Datenträger gespeichert, damit alle ausgeführten Arbeiten reproduziert werden können. Diese Dokumentation, die Konfigurationsdateien, Zertifikate, installierte Software und Scripte sind in der unten ersichtlichen Verzeichnisstruktur abgelegt.