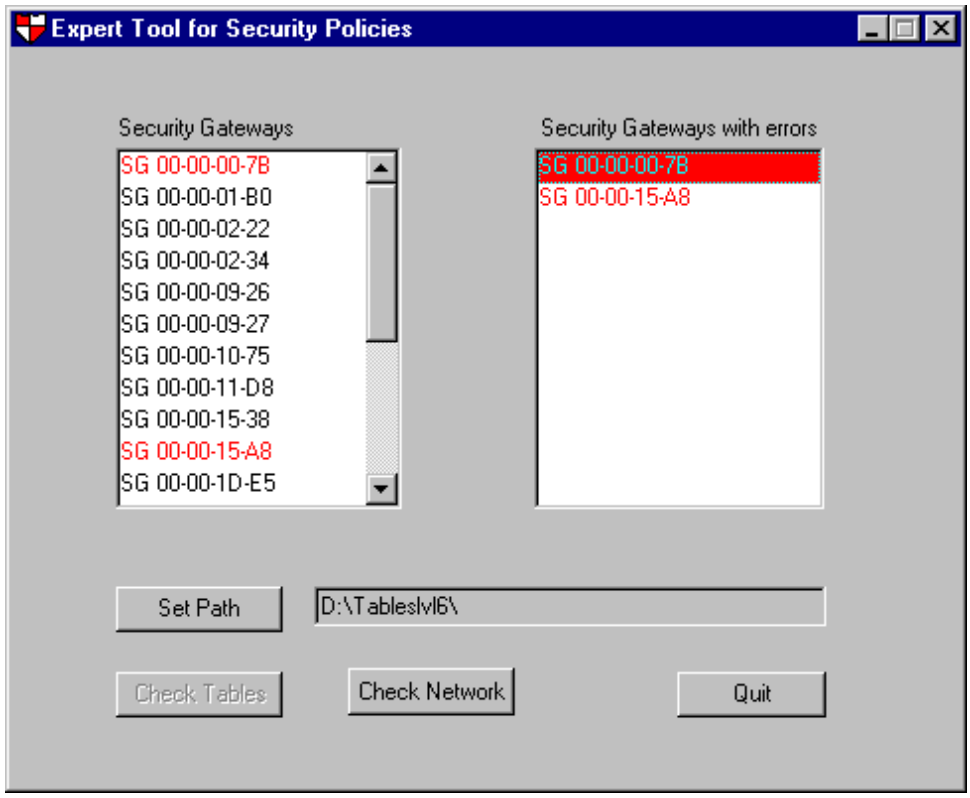
Diplomarbeit
2000

Expert Tool for Security Policies



Cristian D'Aquino

Christoph Meier

Inhaltsverzeichnis

1 Zusammenfassung

Security Gateways bilden die Endpunkte von sicheren IPsec Verbindungen in einem Virtual Private Network (VPN). Ähnlich wie ein Router entscheidet ein Security Gateway auf der Basis von konfigurierbaren Regeln, wie ein ankommendes oder abgehendes IP-Paket behandelt werden soll. Diese Regeln, Security Policies genannt, werden in einer Security Policy Database (SPD) abgespeichert.

In dieser Diplomarbeit hatten wir die Aufgabe, ein Experten-Diagnose-Tool für Security Policies zu konzipieren und anschliessend in Microsoft Visual C++ zu programmieren (inkl. Grafischer Benutzeroberfläche). Dieses Experten-Tool wird von unserem Industriepartner Omnisec AG verwendet werden, um ihre Security Policies zu prüfen, bevor diese in ein Gerät der Produktfamilie Omnisec 41x geladen werden.

Zuerst analysierten wir alle möglichen Fehler und Inkonsistenzen, die bei einer verteilten Network Security Policy (d.h. aufgeteilt in mehrere lokale Security Policies) auftreten können. Danach entwickelten wir zu jedem dieser Probleme entsprechende Lösungsansätze. Diese Lösungsansätze bauten wir später in unser Experten-Tool ein.

Aufbauend auf den von Omnisec AG definierten Schnittstellen entwickelten wir unser Experten-Tool, welches folgende Funktionalität umfasst. Die gesamte SPD wird in unser Programm geladen. Anschliessend wird die SPD auf Fehler und Inkonsistenzen geprüft. Falls Fehler bzw. Inkonsistenzen vorhanden sind, werden diese rot markiert und mit einer Problembeschreibung kommentiert.

Den grössten Teil der Zeit investierten wir in die Programmierung und in das Fehlersuchen bzw. Testen unseres Experten-Tools. Dies ermöglichte uns wertvolle Erfahrungen im Bereich Softwaredesign und Fehlereingrenzung zu sammeln.

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, 29. Oktober 2000

Christoph Meier Cristian D'Aquino

2 Abstract

Security Gateways form the endpoints of secure IPsec connections in a Virtual Private Network (VPN). Similar to a router a Security Gateway decides with configurable rules how to treat an incoming or outgoing IP packet. These rules, called Security Policies, are saved in a Security Policy Database (SPD).

In this degree dissertation we had the task to design an expert diagnose tool for Security Policies and afterwards to program it in Microsoft Visual C++ (inclusive a graphical user interface). This expert tool is going to be used from our industrial partner Omnisec AG to prove their Security Policies before they are loaded into a device of the product family Omnisec 41x.

First we analysed all possible errors and inconsistencies, which can occur in a distributed network Security Policy (i.e. splitted in several local Security Policies). Then we searched for each problem possible solutions. These solutions were later integrated in our expert tool.

Based on the Omnisec interfaces we developed our expert tool, which contains following functionality. The whole SPD is loaded in our program. Afterwards we examine the SPD whether there are any errors or inconsistencies. In the case of errors or inconsistencies, they are marked red and commented with a problem description.

We invested the most time in programing, searching errors and testing our expert tool. This offered us to gain valuable experiences in softwaredesign and localization of errors.

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

Kommunikationssysteme (KSy)

Praktische Diplomarbeiten 2000 - Sna00/5

Expert Tool for Security Policies

Studierende:
- Cristian D'Aquino, IT3a
- Christoph Meier, IT3a

Industriepartner:
- Omnisec AG, Rietstrasse 14, CH-8108 Dällikon (http://www.omnisec.ch)

Termine:
- Ausgabe: Donnerstag, 7.09.2000 15:00 - 17:00 im E509
- Abgabe: Montag, 30.10.2000 12:00

Beschreibung:
Security Gateways bilden die Endpunkte von sicheren IPsec Verbindungen. Ähnlich wie ein Router entscheidet ein Security Gateway auf der Basis von konfigurierbaren Regeln, wie ein ankommendes oder abgehendes IP-Paket behandelt werden soll. Diese Regeln, Security Policies genannt, werden in einer Security Policy Database (SPD) abgespeichert.

In dieser Diplomarbeit soll ein Experten-Diagnose-Tool für Security Policies konzipiert und erstellt werden. Es soll folgende Funktionalität umfassen:

- Die gesamte SPD mehrerer Security Gateways soll als binäre Files in das Experten-Tool geladen werden.
- Das Experten-Tool soll die Funktionalität und Konsistenz der Einträge prüfen.
- Syntaktisch-falsche, doppelte und redundante Einträge sollen als solche erkannt und markiert werden.
- Da der Security Gateway die einzelnen Security Policies nach einer "first match" Methode durchgeht, soll das Tool Vorschläge für die beste Reihenfolge der Regeln generieren, so dass selektivere Policies vor allgemeineren angeordnet werden.

Der Dialog mit dem Anwender soll über eine grafische Oberfläche realisiert werden.

Ziele:
Die Diplomarbeit besteht aus zwei Teilen - einem eher theoretischen Analyse-Teil, sowie einem praktischen Implementations-Teil.

- **Teil 1 - Analyse**
  Es ist aufzuzeigen, welche Fehler und Inkonsistenzen bei einer "verteilten" Network Security Policy (d.h. aufgesplittet in mehrere lokale Security Policies) auftreten können und wie diese Probleme mit einem Experten Tool erkannt werden können.

  Nachfolgend eine nicht-vollständige Liste möglicher Problemfälle, welche die Sicherheit und Konsistenz der Security Policies betreffen und entsprechend erkannt werden müssen:
  - Syntax der einzelnen Security Policy Einträge
  - Konsistenz (doppelte, sowie redundante Einträge, doppelte IP Adressen, überlappende Netze/Subnetze, etc)

  Da ein Netzwerk bis zu 5000 Security Gateways umfassen kann, soll eine optimale Architektur der Policy Database erarbeitet werden.

- **Teil 2 - Implementation**
  Basierend auf den Erkenntnissen der Analyse ist eine Experten-Software mit folgenden Eigenschaften zu entwicklen:
  - Einlesen der lokalen Security Policies mehrerer Security Gateways als Files im Binärformat. Die Struktur der Security Policies soll auf dem Omnisec Table Model basieren.
  - Zusammenfassung aller lokalen Security Policies zu einer globalen Network Security Policy.
  - Implementierung einer zu definierenden Teilmenge der im Analyse-Teil definierten Problem- und Fehler-Detektionsalgorithmen.
  - Realisierung folgender Grundansichten der Network Security Policy:
    - Klarverbindungen
    - Chiffrierte Verbindungen
    - Keine Verbindung (Verbindung verboten oder keine gemeinsamen Proposals)
    - Problemverbindungen (Mehrfachdefinitionen, Syntaxfehler, etc.)
    - Auf obigen Ansichten sollen Filter nach Problemkategorien möglich sein.
    - Die verschiedenen Ansichten müssen auf einen Drucker ausgegeben werden können.
    - Sortieren der Security Policies nach "closest match"
    - Exportieren der einzelnen modifizierten Security Policies zum entsprechenden Security Gateway.

Das Experten-Tool soll als objektorientierte C++ Applikation unter Windows NT mit Hilfe der Entwicklungsumgebung "Microsoft Visual Studio" erstellt werden.
Das "Graphical User Interface" und die Kommentare im "Source Code" sollen in englischer Sprache erstellt werden. Die schriftliche Dokumentation der Diplomarbeit kann in deutscher Sprache abgefasst werden.

Infrastruktur / Tools:

- Raum: **E523**
- Rechner: 2 PCs mit Windows NT 4.0
- SW-Tools: Microsoft Visual Studio

Literatur / Links:

- Omnisec Spezifikation "IPSEC Parameter Storage"
- SSH IPSEC Express Toolkit 3.0.1 Installation and Porting Guide
  Chapter 2 - Configuring SSH IPSEC (paper copy only)
- IETF RFC 2401
  Security Architecture for the Internet Protocol
- IETF IP Security Policy Workgroup, several drafts:
  http://www.ietf.org/html.charters/ipsp-charter.html
  - IPSP Requirements
    http://www.ietf.org/internet-drafts/draft-ietf-ipsp-requirements-00.txt
  - IPsec Policy Architecture
    http://www.ietf.org/internet-drafts/draft-ietf-ipsp-arch-00.txt
  - IPsec Configuration Policy Model
    http://www.ietf.org/internet-drafts/draft-ietf-ipsp-config-policy-model-01.txt
  - Security Policy Specification Language
    http://www.ietf.org/internet-drafts/draft-ietf-ipsp-spsl-00.txt
  - Security Policy Protocol
    http://www.ietf.org/internet-drafts/draft-ietf-ipsp-spp-00.txt
- IETF RFC 2407
  The Internet IP Security Domain of Interpretation for ISAKMP
- IETF RFC 2408
  Internet Security Association and Key Management Protocol (ISAKMP)
- IETF RFC 2409
  The Internet Key Exchange (IKE)

Winterthur, 7. September 2000

Dr. Andreas Steffen

4 Zeitplan

4.1 Soll-Zeitplan

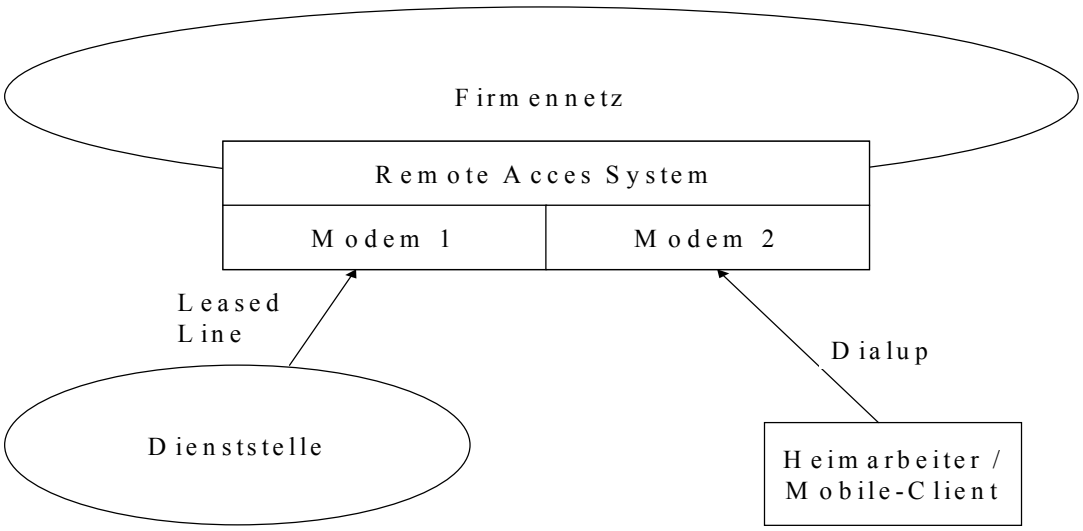
4.2 Ist-Zeitplan

5 Allgemeines über VPN

Ein virtuelles privates Netz verbindet lokale Netze über öffentliche Netze miteinander. Der Ausdruck „privat“ bedeutet, dass die Verbindung zwischen zwei Rechnern genauso gut gesichert ist, wie wenn sie zusammen in einem lokalen Netzwerk 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 Netzwerk verbindet. Angesichts dieser Gründe folgte der Name „Virtual Privat Network“.
Wie erwähnt, werden die Rechner über ein unsicheres öffentliches Netz verbunden, trotzdem sind durch VPN folgende Sicherheitsvoraussetzungen für eine gesicherte Verbindung gegeben:

- Authentifizierung des Kommunikationspartners
- Integrität der Information, d.h. die gesendeten und empfangenen Daten sind nicht verändert worden
- Abhörsicherheit durch Verschlüsselung
- Identitätsverbergung der Kommunikationspartner
- 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 können. Die Skizze soll einen Überblick über die alte Methode des Anschlusses an das Firmennetz illustrieren.



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

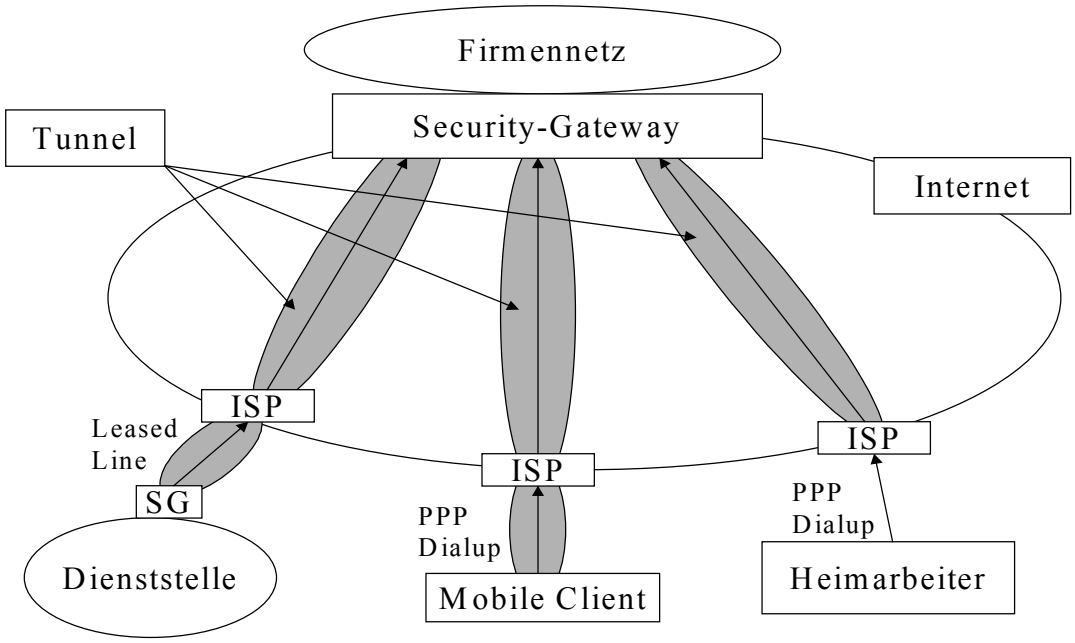
Verbindungen mittels Mietleitungen zu eigenen lokalen Satellitenbüronetzen mit dem Firmennetz sind trotz steigendem Konkurrenzkampf der Leitungsanbieter eine

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 Heimarbeitern und mobile Clients (Ferngespräche bezahlt man jetzt unter Nutzung des Internets zum Ortstarif).
- Einsparungen von 20-50% der Kosten von Mietleitungen.
- Weltweite Zugriffsmöglichkeiten aufs Internet und somit auf das lokale Firmennetz.
- Weniger Hardware und somit weniger Unterhaltsarbeit im Firmennetz, also kein Remote Access System mit seinen Modems.

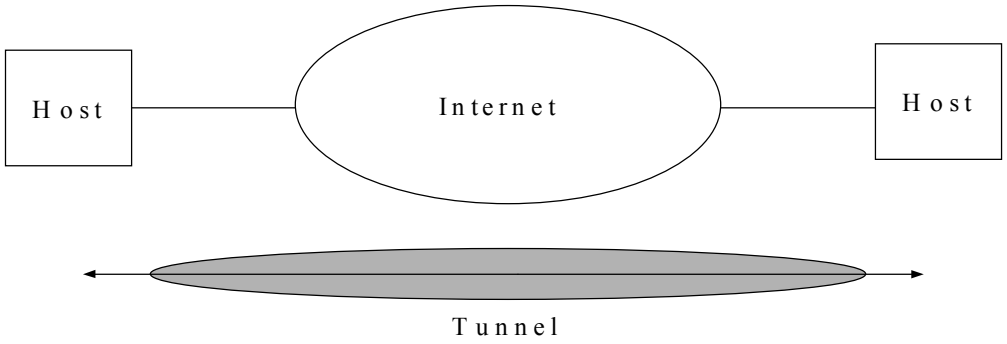
Die Skizze illustriert einige VPN-Lösungen übers Internet



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

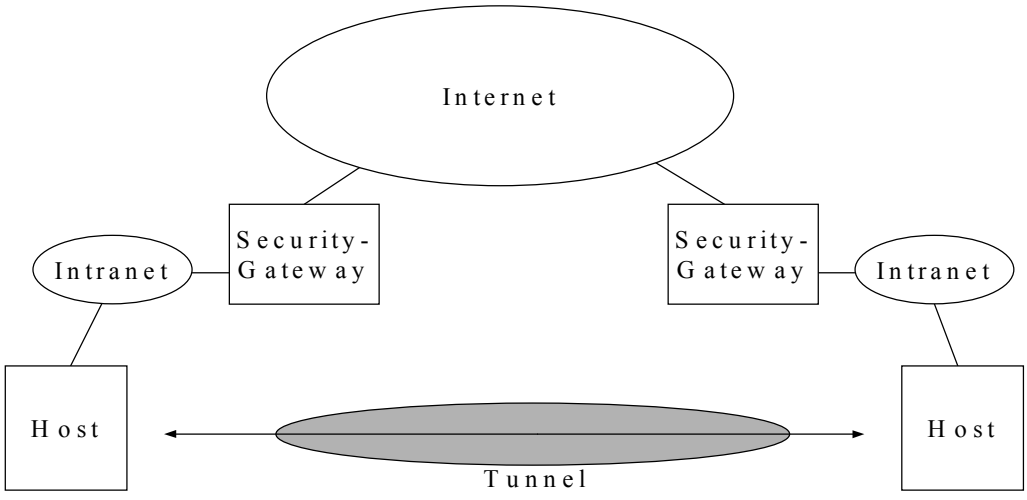
5.1 VPN Architekturen

End-to-End



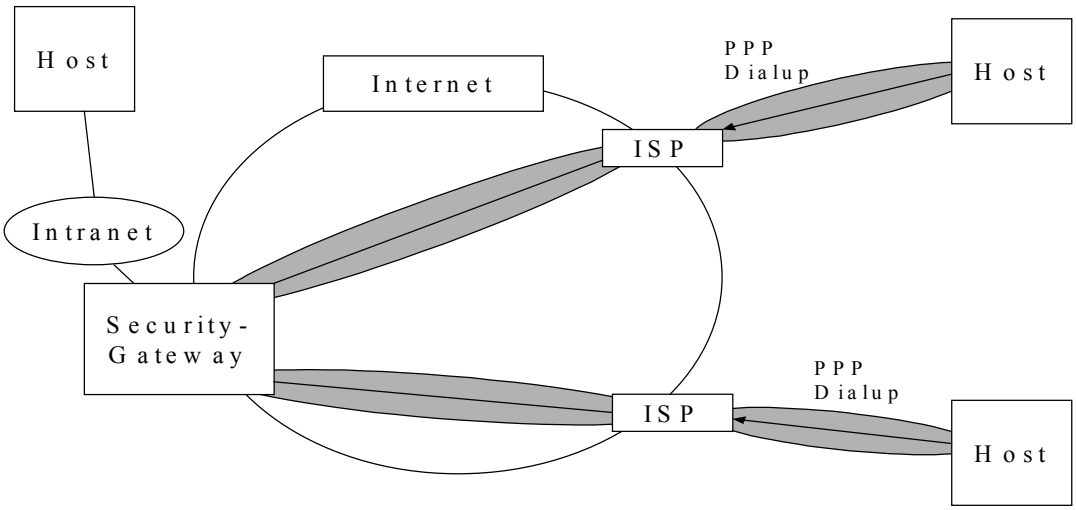
Diese ist die sicherste Lösung für ein VPN-Verbindung über das Internet. Der Tunnel mit den verschlüsselten Daten deckt die ganze Verbindung bis zu den Hosts ab. Dadurch kann eine Angriff auf der ganzen Verbindungslänge ausgeschlossen werden. Dazu muss jeder an der verschlüsselten Kommunikation beteiligter 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.

Site-to-Site



Bei Site-to-Site tauschen zwei Intranetze mit ihren Stationen Daten übers Internet aus. 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 überlassen werden. Damit überträgt man den administrativen Aufwand und den Support dem Provider. Aus sicherheitstechnischen Gründen ist das sicher nicht eine optimale Lösung.

End-to-Site



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 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.

5.2 Was für VPN-Protokolle gibt es?

Da das ursprüngliche *TCP/IP-Referenzmodell* des Internets keinen Sicherheitsaspekt bietet, musste es 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-Referenzmodell	Sicherheits-Protokolle	Kurze Beschreibung
Applikationsschicht	IPsec (IKE)	IP Internet Security (Internet Key Exchange)
	S-HTTP	Secure Hyper Text Transfer Protocol Sichere Übertragung von WWW-Seiten
	S-MIME	Secure Multipurpose Internet Mail Extension Standard zur sicheren Übertragung von Email
TCP/UDP Transportschicht	SOCKS	Socket Secure Server, Standard zur Nutzung von Internet-Diensten über einen Firewall
	SSL	Secure Sockets Layer, Netscapes Technik zur sicheren Übertragung von HTTP (Hyper Text Transfer Protocol)
IP Vermittlungsschicht	IPsec(AH,ESP)	IP Internet Security
	Paket filtering	Firewall
Sicherungsschicht	PAP	Layer 2 Tunneling Protocol Password Authentication Protocol (PAP)
	CHAP	Challenge Handshake Authentication Protocol
Bitübertragungsschicht		

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.

5.3 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.

5.3.1 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 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.

5.3.2 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 ursprüngliche 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. Ein weiteres Plus von Tunneling ist die Benützung von *privaten IP-Adresse*n. In der Regel ist das lokale Netz mit *privaten IP-Adressen* aufgebaut.

5.3.3 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 Sicherheitsanforderungen 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

5.3.4 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

5.3.5 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 Verschlüsselungsalgorithmus nichts. Genau diese Sicherheit wird durch IKE geboten, wenigstens was die Erzeugung anbelangt.

6 Pflichtenheft

In dieser Diplomarbeit soll ein Experten-Diagnose-Tool für Security Policies konzipiert und erstellt werden. Die Arbeit besteht aus zwei Teilen - einem eher theoretischen Analyse-Teil, sowie einem praktischen Implementations-Teil.

Teil 1 - Analyse
Es ist aufzuzeigen, welche Fehler und Inkonsistenzen bei einer "verteilten" Network Security Policy (d.h. aufgesplittet in mehrere lokale Security Policies) auftreten können und wie diese Probleme mit einem Experten Tool erkannt werden können.
Nachfolgend eine nicht-vollständige Liste möglicher Problemfälle, welche die Sicherheit und Konsistenz der Security Policies betreffen und entsprechend erkannt werden müssen:

- Syntax der einzelnen Security Policy Einträge
- Konsistenz (doppelte, sowie redundante Einträge, doppelte IP Adressen, überlappende Netze/Subnetze, etc.)

Da ein Netzwerk bis zu 5000 Security Gateways umfassen kann, soll eine optimale Architektur der Policy Database erarbeitet werden.

Teil 2 - Implementation
Basierend auf den Erkenntnissen der Analyse ist eine Experten-Software mit folgenden Eigenschaften zu entwickeln:

- Einlesen der lokalen Security Policies mehrerer Security Gateways als Files im Binärformat. Die Struktur der Security Policies soll auf dem Omnisec Table Model basieren.
- Implementierung einer zu definierenden Teilmenge der im Analyse-Teil definierten Problem- und Fehler-Detektionsalgorithmen.
- Realisierung folgender Fehleransichten der Network Security Policy:
  - Fehlerhafte Tabellen
  - Fehlerhafte Tabelleneinträge
  - Fehlerhafte Verbindungen
  - Fehlerhafte Verbindungseinträge
  - Gesamtübersicht aller fehlerhaften Security Gateways

Das Experten-Tool soll als objektorientierte C++ Applikation unter Windows NT mit Hilfe der Entwicklungsumgebung "Microsoft Visual Studio" erstellt werden.

Das "Graphical User Interface" und die Kommentare im "Source Code" sollen in englischer Sprache erstellt werden. Die schriftliche Dokumentation der Diplomarbeit kann in deutscher Sprache abgefasst werden.

Winterthur, 8. September 2000

Christoph Meier Cristian D'Aquino

7 Problemanalyse

Das Problem kann in zwei Teilprobleme unterteilt werden. Zuerst betrachten wir die Tabellen eines einzelnen Security Gateways, danach werden dann die Tabellen aller Security Gateways miteinander verglichen.

7.1 Tabellen eines Security Gateways

Wir haben dieses Problem wiederum in 4 Teilprobleme aufgeteilt. Zuerst wird nur ein einzelner Eintrag einer Tabelle überprüft. Als zweites werden die einzelnen Tabellen untersucht. Danach werden alle Indizes überprüft und am Schluss werden die Tabellen untereinander überprüft.

7.1.1 Einzelne Einträge (Stufe 1)

(der Aufbau der Tabellen ist in 10.2.2.1 grafisch dargestellt)

Node Table Entry
Bei diesen Einträgen kann man nur die IP- Adresse überprüfen. Die Adresse muss eine öffentliche Adresse sein, welche entweder eine Class A, B oder C Adresse ist. Sie darf auch keine speziell verwendete Adresse sein (Bsp. 127.0.0.1, ist eine öffentliche Class A Adresse, welche jedoch als „local loopback“ verwendet wird).

Network Table Entry
Auch hier müssen die IP-Adressen (range start und range end) gültige sein (wie bei Node Table, jedoch können sie auch eine private Adresse sein). Weiter muss die Start- Adresse tiefer sein als die End- Adresse und es dürfen keine ungültigen Adressen dazwischen liegen (Bsp. Start- Adresse = 32.4.2.6, End- Adresse = 131.34.2.5 ist nicht gültig, da der Bereich des „local loopback“ dazwischen liegt). Der Index des Security Gateways (Node Index) muss auf einen Security Gateway zeigen (d.h. er darf nicht 0 sein, obere Grenze wird hier noch nicht überprüft).

Connection Table Entry
Bei diesen Einträgen muss die Protokoll ID gültig sein (siehe Typendefinition: SshInetIPProtocolID in Datei: oagtypes.hpp), die Indizes (local / remote network, IkeProposal und IpSecProposal) müssen auf einen Eintrag zeigen (d.h. sie dürfen nicht 0 sein, obere Grenze wird hier noch nicht überprüft) und der local- und remote-Networkindizes dürfen nicht gleich sein.

Lifetime Table Entry
Bei diesen Einträgen müssen die Werte hard_seconds und hard_kbytes grösser als 0 und grösser als die Werte soft_secondes und soft_kbyte sein.

IKE Transform Table Entry
Die Encryption und Hash Algorithmen müssen gültig sein. (siehe Typendefinition in oagtypes.hpp)

ESP Transform Table Entry
Die Encryption und Mac Algorithmen sowie die Mode müssen gültig sein. (siehe Typendefinition in oagtypes.hpp)

AH Transform Table Entry
Der Mac Algorithmus und die Mode müssen gültig sein. (siehe Typendefinition in oagtypes.hpp)

Comp Transform Table Entry
Diese Tabelle existiert noch nicht

IKE Proposal Table Entry
Bei der jetzigen Version der Security Gateways müssen die Felder group und authenticationMethode 0 bzw. 1 sein. Der Index auf die Lifetime- Tabelle muss auf einen Eintrag zeigen (d.h. Er darf nicht 0 sein, obere Grenze kann hier noch nicht bestimmt werden). Bei dem Feld Transform muss mindestens der erste Wert auf einen Eintrag in der IKE Transform Table zeigen (d.h. Er darf nicht 0 sein (die weiteren Einträge dürfen 0 sein), obere Grenze wird hier noch nicht überprüft).

IPsec Proposal Table Entry
Der Index auf die Lifetime- Tabelle muss auf einen Eintrag zeigen (d.h. Er darf nicht 0 sein, obere Grenze kann hier noch nicht bestimmt werden). Die ersten Werte der Felder espTransform, ahTransform und compTransform müssen auf einen Eintrag zeigen (d.h. Er darf nicht 0 sein (die weiteren Einträge dürfen 0 sein), obere Grenze wird hier noch nicht überprüft).

7.1.2 Ganze Tabelle (Stufe 2)

Node Table
Einträge dürfen nicht mehrfach vorhanden sein. Die IP- Adresse und die OAD (Eindeutige Identifikations-Nummer der Security Gateways) dürfen auch nicht mehrfach vorhanden sein.

Network Table
Einträge dürfen nicht mehrfach vorhanden sein. Die einzelnen Netzwerke dürfen sich nicht überschneiden (Ausnahme sind Netzwerke mit Privaten Adressen, welche zu verschiedenen Security Gateways gehören). Subnetze sind erlaubt (müssen gleichen Security Gateway haben, Ausnahme sind Private Netze, welche auch unterschiedlich Security Gateways haben dürfen). Beim überprüfen der Netzwerktabelle muss auch noch geprüft werden, dass ein Security Gateway entweder nur öffentliche- oder nur private- Netzwerke besitzt.

Connection Table
Einträge dürfen nicht mehrfach vorhanden sein. Falls für eine Verbindung zwischen zwei Netzwerken für die einzelnen Protokolle (Protokoll ID) oder für bestimmte Ports (local / remote Port) andere Policies verwendet werden, müssen die Einträge mit der restriktiveren Bestimmung zuerst in der Tabelle stehen (werden nach First-Match ausgewählt. z.B. muss Eintrag für Verbindung mit Port 50 vor dem Eintrag mit „any Port“ stehen).

IKE Transform, AH Transform, ESP Transform, Lifetime, IKE Proposal, IPsec Proposal- Tables
Einträge dürfen nicht mehrfach vorhanden sein. Da bei diesen Tabellen ein doppelter Eintrag keine Sicherheitsprobleme darstellt, haben wir diese Tests aus Zeitgründen weglassen müssen.

7.1.3 Indizes überprüfen (Stufe 3)

Alle Indizes dürfen nicht grösser sein, als die Anzahl Einträge der Entsprechenden Tabellen.

7.1.4 Tabellen Untereinander (Stufe 4)

Die IP- Adressen der Security Gateways dürfen nicht in einem Netzwerk enthalten sein.

Bei einer Verbindung darf das Start und das Zielnetzwerk nicht hinter dem gleichen Security Gateway liegen.

Fall Subnetze vorhanden sind, müssen die Verbindungen für diese Netze zuerst in der Tabelle stehen.

Löcher in der Konfiguration (Netzwerke, Security Gateways zu welchen es keine Verbindungen gibt) sollen erkannt werden. Da dies jedoch keine Sicherheitsprobleme darstellen, haben wir diesen Test aus Zeitgründen weggelassen.

7.2 Tabellen aller Security Gateways

Auch dieses Problem haben wir in zwei Teilprobleme aufgeteilt. Zuerst die einzelnen Tabellen mit ihrem Gegenstück der anderen Security Gateways, danach alle Tabellen von allen Security Gateways.

7.2.1 Tabellen miteinander (Stufe 5)

Node Table
Einträge mit gleicher OAD müssen auch die gleiche IP- Adresse haben. Das gleiche gilt auch für Einträge mit gleicher IP- Adresse (OAD muss auch gleich sein).

Network Table
Netzwerke welche in den verschiedenen Security Gateways anders definiert sind (z.B. anstelle eines Grossen, 4 Kleine), sind verdächtig und sollten erkannt werde. Aus Zeitgründen konnten wir dies nicht Implementieren.
Ein Netzwerk muss in jeder Tabelle zu dem gleichen Security Gateway gehören. Aus Zeitgründen konnten wir dies nicht Implementieren.

Die anderen Tabellen sind unproblematisch.

7.2.2 Alle Tabellen miteinander (Stufe 6)

Jede Verbindung benötigt auch einen Eintrag im Ziel Security Gateway.
Die beiden Einträge für jede Verbindung müssen gemeinsame Verschlüsselungsverfahren, Lifetime, usw. aufweisen.

Die IP- Adressen der Security Gateways dürfen in keinem Netzwerk enthalten sein. Konnten wir aus Zeitgründen nicht Implementieren (falls alle Security Gateways die gleichen Node Table und Network Table besitzen, genügt der Test auf Stufe 4)

8 Lösungskonzept

Die Reihenfolge der Einzelnen Tests haben wir von der Reihenfolge in der Problemanalyse übernommen. Obwohl dieses Vorgehen erheblich mehr Zeit benötigt (viel mehr IO auf Festplatte), hat es den grossen Vorteil, dass es übersichtlicher ist. So werden bei uns zuerst alle Einträge auf Gültigkeit geprüft. Danach werden die einzelnen Tabellen geprüft, usw. Wir haben uns auch dafür entschieden, viele kleine Funktionen anstelle von wenigen grossen Funktionen zu verwenden, da viele kleine Probleme einfacher zu lösen sind, als ein grosses Problem.

Im Kapitel 10 Software wird auf das Softwaredesign genauer eingegangen.

9 Infrastruktur

9.1 Software

- Microsoft Visual C++ 6.0
- Rational Rose 98
- CodeVizor Version 2.0
- Windows NT 4.0 Workstation

9.2 Hardware

- Prozessor: Pentium III mit 450 MHz
- Motherboard: ASUS P2B-S
- Arbeitsspeicher: 128 MB
- Grafikkarte: Matrox Millenium G400 32MB

10 Software

10.1 Einleitung

Wir werden im Kapitel Software auf drei Programme eingehen:
1. "IpTables" (von Omnisec AG geliefertes Programm)
2. "ExpertTool for Security Policies" (von uns erstelltes Programm)
3. "IPsecTables" (von uns erstelltes Hilfsprogramm)

Da unser Programm "ExpertTool for Security Policies" fast alle Quellcode-Dateien von "IpTables" (ausser main.cpp) einbindet und gebraucht, werden wir zuerst ausführlich auf das Programm "IpTables" eingehen. Wenn man dieses Programm versteht, ist es nicht mehr schwierig zu begreifen, wie das Programm "ExpertTool for Security Policies" funktioniert, welches jedoch wesentlich umfangreicher ist.

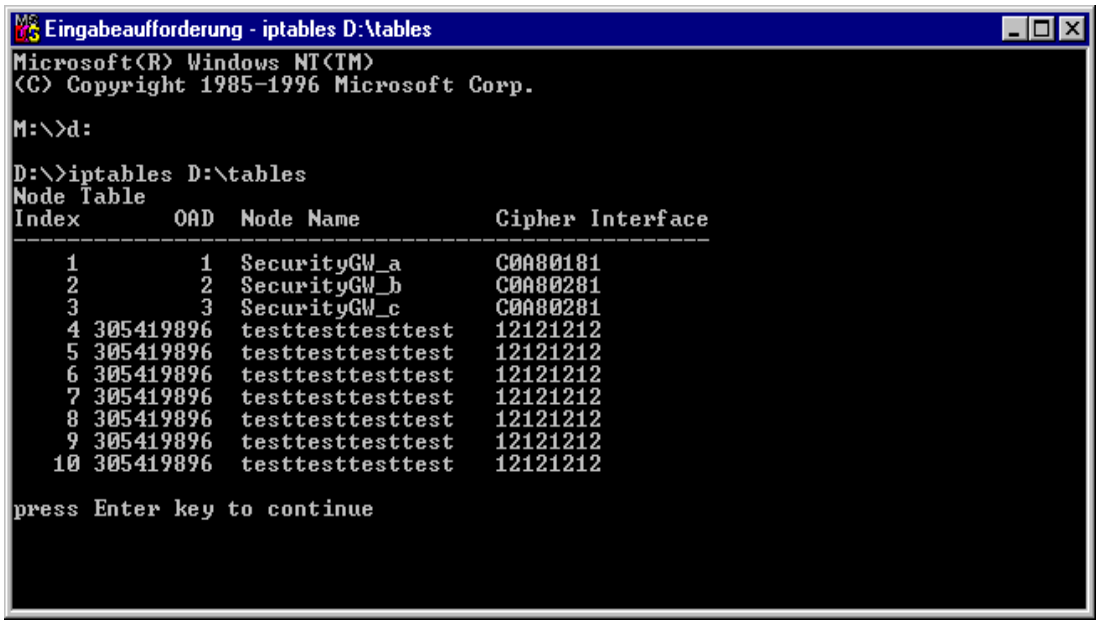
10.2 Programm "IpTables"

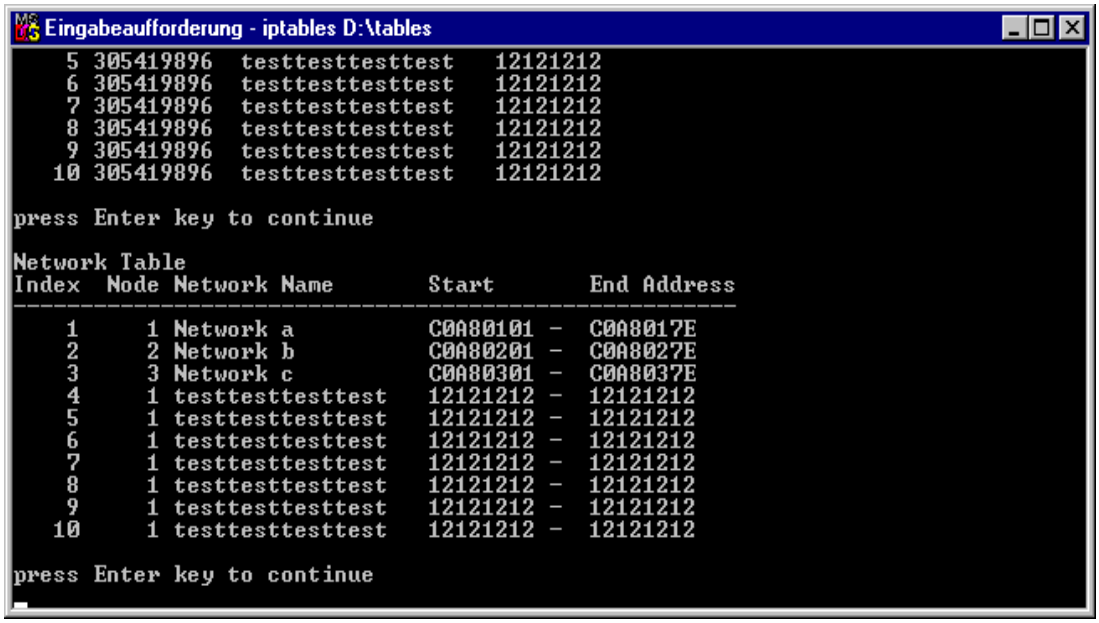
10.2.1 Benutzeranleitung

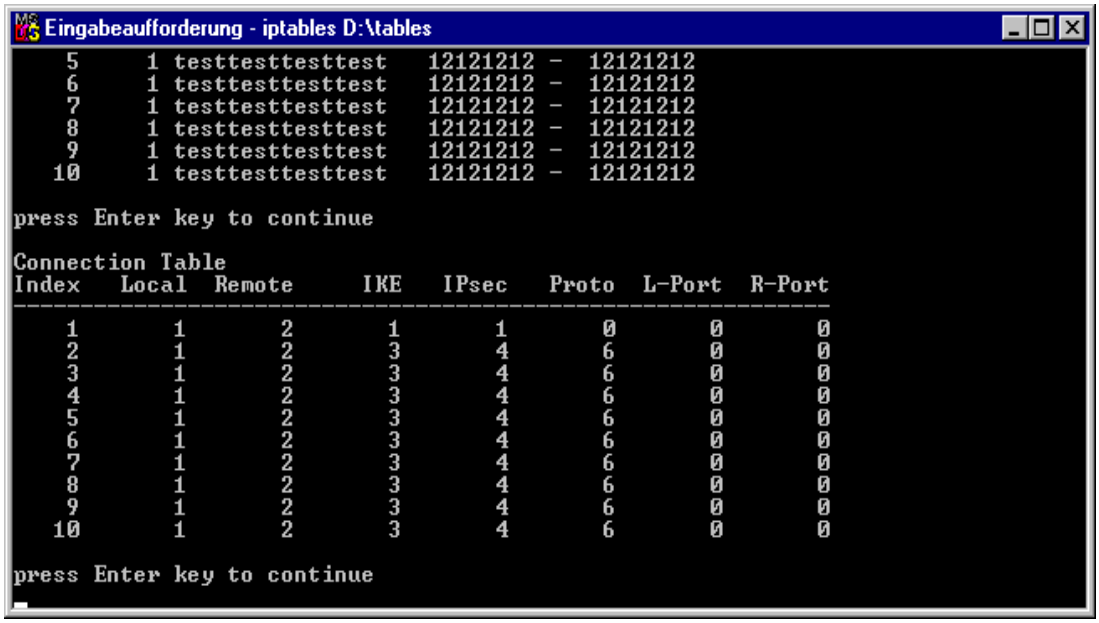
Von der Firma Omnisec haben wir ein fertiges Programm namens „iptables.exe“ erhalten, welches folgende 9 Dateien erstellt:

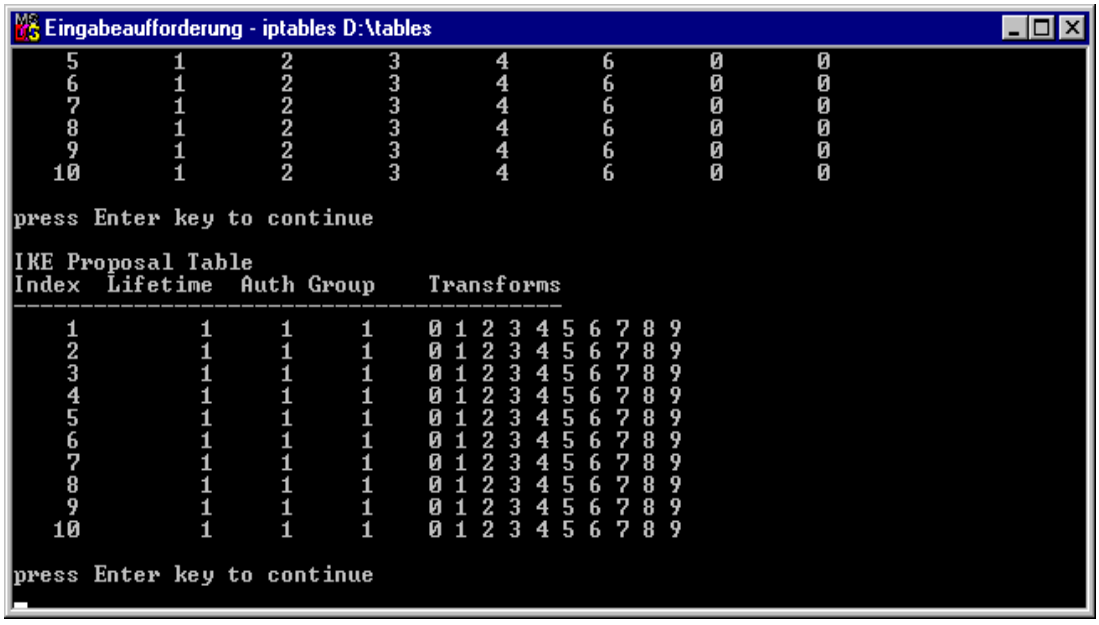
- node.tab
- network.tab
- connection.tab
- ikeproposal.tab
- ipsecproposal.tab
- iketransform.tab
- esptransform.tab
- ahtransform.tab
- lifetime.tab

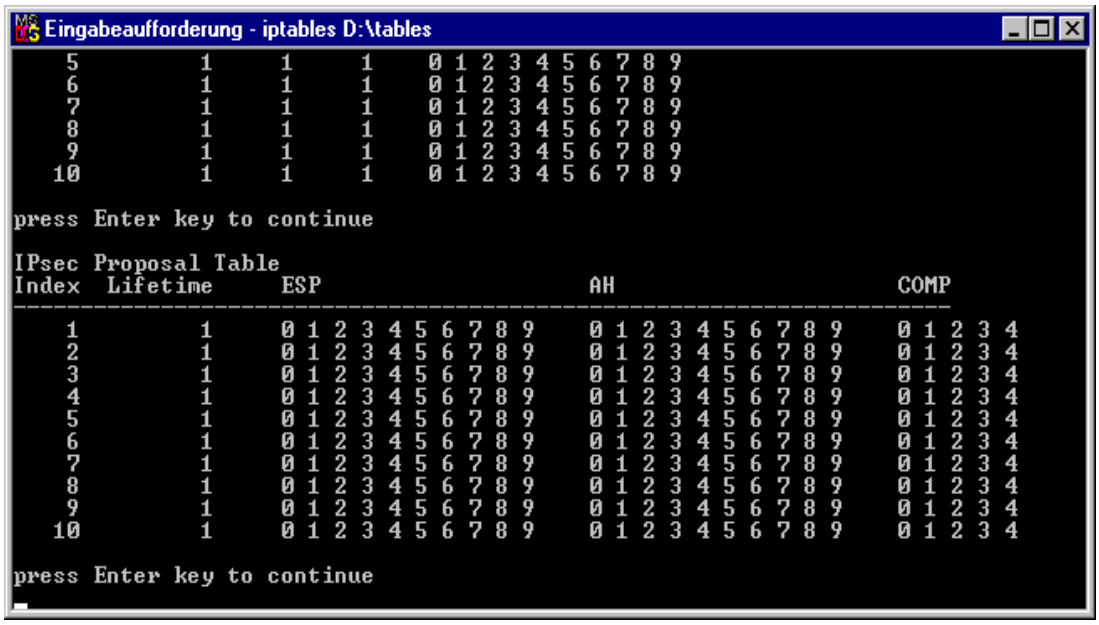
Dieses Programm ist eine Konsolen Applikation (DOS-Fenster).
Es wird mit folgendem Befehl gestartet: **iptables <Pfad>**
Beispiel: **iptables D:\tables**
Im angegeben Pfad werden die 9 Dateien gespeichert (Achtung! Pfad muss vorhanden sein; in unserem Beispiel muss also der Ordner "tables" auf dem Laufwerk D von Hand erstellt werden). Jede dieser Dateien speichert eine Tabelle (Sinn und Zweck ist im Kapitel 10.2.2.1 erklärt). Die Tabellen sind in diesem Programm "hart kodiert", d.h. man kann sie nicht editieren. Falls sie schon vorhanden sind, werden sie nicht gelöscht, sondern es werden die gleichen Einträge nochmals hinzugefügt.
Wenn man das Programm laufen lässt werden die Tabellen im DOS-Fenster angezeigt. Zur Veranschaulichung sind hier noch die Screenshots der einzelnen Tabellen dargestellt:

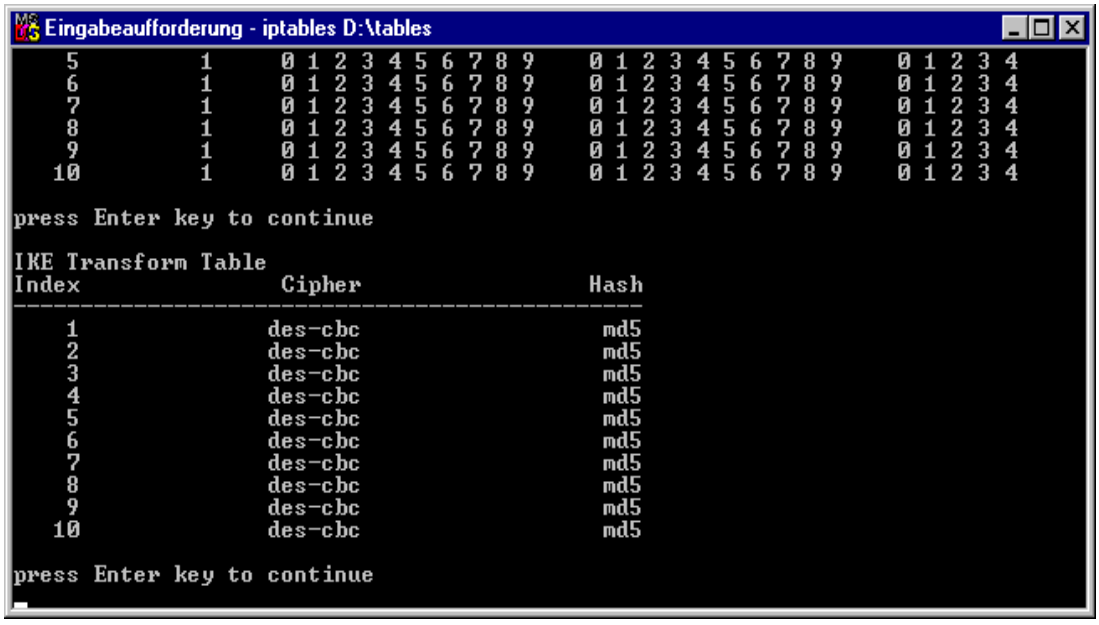


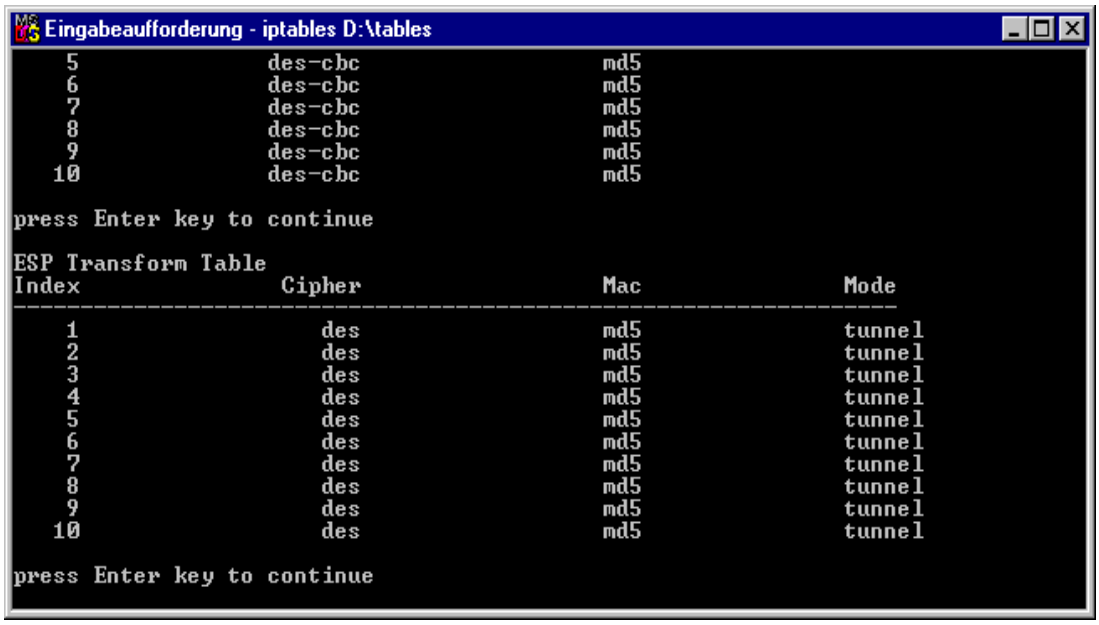

Eingabeaufforderung - iptables D:\tables
5 305419896 testtesttesttest 12121212
6 305419896 testtesttesttest 12121212
7 305419896 testtesttesttest 12121212
8 305419896 testtesttesttest 12121212
9 305419896 testtesttesttest 12121212
10 305419896 testtesttesttest 12121212
press Enter key to continue
Network Table
Index Node Network Name Start End Address
1 1 Network a C0A80101 - C0A8017E
2 2 Network b C0A80201 - C0A8027E
3 3 Network c C0A80301 - C0A8037E
4 1 testtesttesttest 12121212 - 12121212
5 1 testtesttesttest 12121212 - 12121212
6 1 testtesttesttest 12121212 - 12121212
7 1 testtesttesttest 12121212 - 12121212
8 1 testtesttesttest 12121212 - 12121212
9 1 testtesttesttest 12121212 - 12121212
10 1 testtesttesttest 12121212 - 12121212
press Enter key to continue

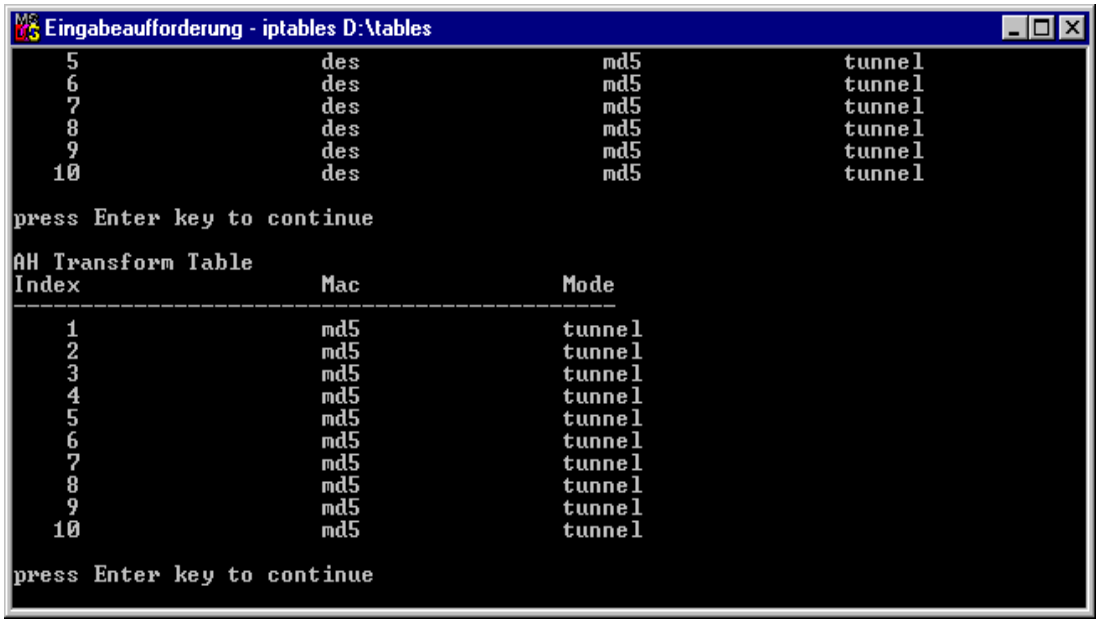

Eingabeaufforderung - iptables D:\tables
5 1 testtesttesttest 12121212 - 12121212
6 1 testtesttesttest 12121212 - 12121212
7 1 testtesttesttest 12121212 - 12121212
8 1 testtesttesttest 12121212 - 12121212
9 1 testtesttesttest 12121212 - 12121212
10 1 testtesttesttest 12121212 - 12121212
press Enter key to continue
Connection Table
Index Local Remote IKE IPsec Proto L-Port R-Port
1 1 2 1 1 0 0 0
2 1 2 3 4 6 0 0
3 1 2 3 4 6 0 0
4 1 2 3 4 6 0 0
5 1 2 3 4 6 0 0
6 1 2 3 4 6 0 0
7 1 2 3 4 6 0 0
8 1 2 3 4 6 0 0
9 1 2 3 4 6 0 0
10 1 2 3 4 6 0 0
press Enter key to continue

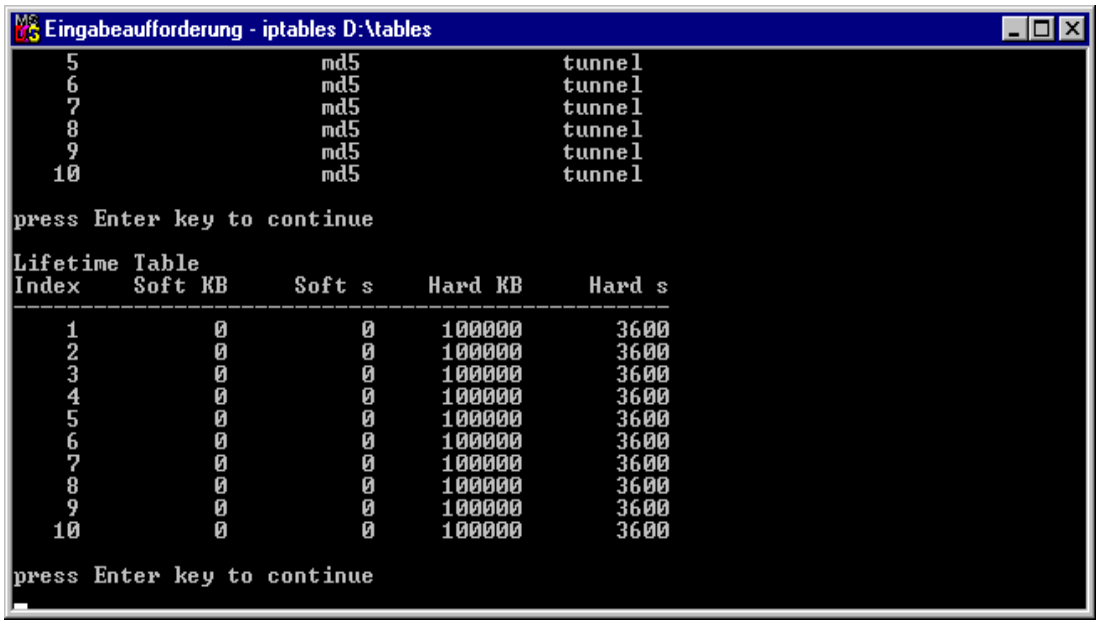

Eingabeaufforderung - iptables D:\tables
5 1 2 3 4 6 0 0
6 1 2 3 4 6 0 0
7 1 2 3 4 6 0 0
8 1 2 3 4 6 0 0
9 1 2 3 4 6 0 0
10 1 2 3 4 6 0 0
press Enter key to continue
IKE Proposal Table
Index Lifetime Auth Group Transforms
1 1 1 1 0 1 2 3 4 5 6 7 8 9
2 1 1 1 0 1 2 3 4 5 6 7 8 9
3 1 1 1 0 1 2 3 4 5 6 7 8 9
4 1 1 1 0 1 2 3 4 5 6 7 8 9
5 1 1 1 0 1 2 3 4 5 6 7 8 9
6 1 1 1 0 1 2 3 4 5 6 7 8 9
7 1 1 1 0 1 2 3 4 5 6 7 8 9
8 1 1 1 0 1 2 3 4 5 6 7 8 9
9 1 1 1 0 1 2 3 4 5 6 7 8 9
10 1 1 1 0 1 2 3 4 5 6 7 8 9
press Enter key to continue


Eingabeaufforderung - iptables D:\tables
5 1 1 1 0 1 2 3 4 5 6 7 8 9
6 1 1 1 0 1 2 3 4 5 6 7 8 9
7 1 1 1 0 1 2 3 4 5 6 7 8 9
8 1 1 1 0 1 2 3 4 5 6 7 8 9
9 1 1 1 0 1 2 3 4 5 6 7 8 9
10 1 1 1 0 1 2 3 4 5 6 7 8 9
press Enter key to continue
IPsec Proposal Table
Index Lifetime ESP AH COMP
1 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
2 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
3 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
4 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
5 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
6 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
7 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
8 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
9 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
10 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
press Enter key to continue

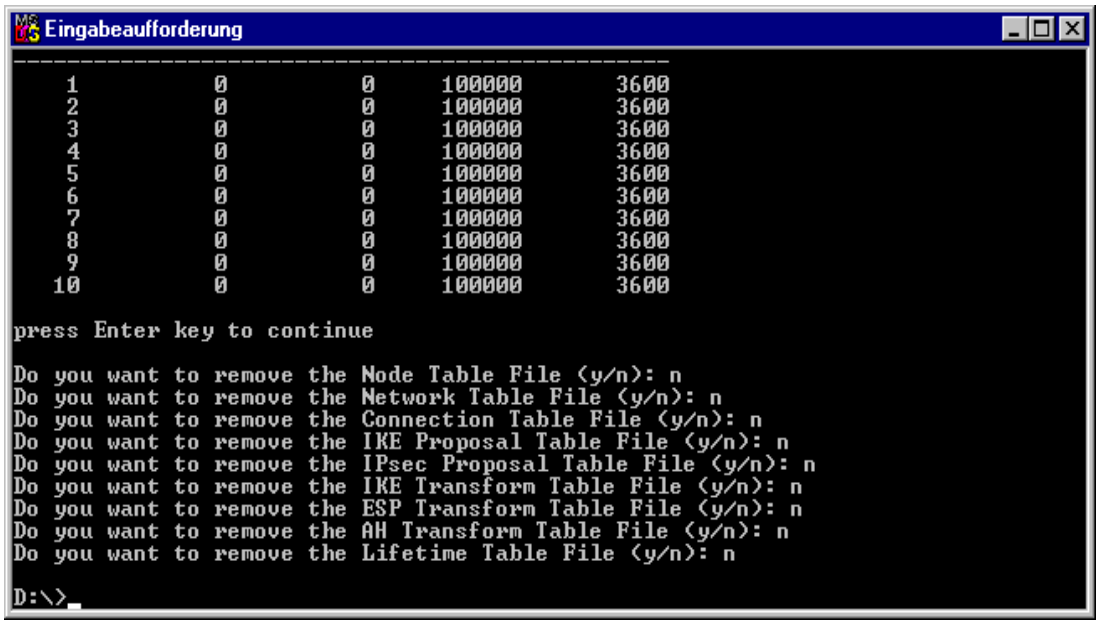

Eingabeaufforderung - iptables D:\tables
5 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
6 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
7 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
8 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
9 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
10 1 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4
press Enter key to continue
IKE Transform Table
Index Cipher Hash
1 des-cbc md5
2 des-cbc md5
3 des-cbc md5
4 des-cbc md5
5 des-cbc md5
6 des-cbc md5
7 des-cbc md5
8 des-cbc md5
9 des-cbc md5
10 des-cbc md5
press Enter key to continue


Eingabeaufforderung - iptables D:\tables
5 des-cbc md5
6 des-cbc md5
7 des-cbc md5
8 des-cbc md5
9 des-cbc md5
10 des-cbc md5
press Enter key to continue
ESP Transform Table
Index Cipher Mac Mode
1 des md5 tunnel
2 des md5 tunnel
3 des md5 tunnel
4 des md5 tunnel
5 des md5 tunnel
6 des md5 tunnel
7 des md5 tunnel
8 des md5 tunnel
9 des md5 tunnel
10 des md5 tunnel
press Enter key to continue


Eingabeaufforderung - iptables D:\tables
5 des md5 tunnel
6 des md5 tunnel
7 des md5 tunnel
8 des md5 tunnel
9 des md5 tunnel
10 des md5 tunnel
press Enter key to continue
AH Transform Table
Index Mac Mode
1 md5 tunnel
2 md5 tunnel
3 md5 tunnel
4 md5 tunnel
5 md5 tunnel
6 md5 tunnel
7 md5 tunnel
8 md5 tunnel
9 md5 tunnel
10 md5 tunnel
press Enter key to continue


Eingabeaufforderung - iptables D:\tables
5 md5 tunnel
6 md5 tunnel
7 md5 tunnel
8 md5 tunnel
9 md5 tunnel
10 md5 tunnel
press Enter key to continue
Lifetime Table
Index Soft KB Soft s Hard KB Hard s
1 0 0 100000 3600
2 0 0 100000 3600
3 0 0 100000 3600
4 0 0 100000 3600
5 0 0 100000 3600
6 0 0 100000 3600
7 0 0 100000 3600
8 0 0 100000 3600
9 0 0 100000 3600
10 0 0 100000 3600
press Enter key to continue

Am Schluss fragt das Programm, ob die Tabellen wieder gelöscht werden sollen.



10.2.2 Softwaredesign

10.2.2.1 Sinn und Zweck der Tabellen

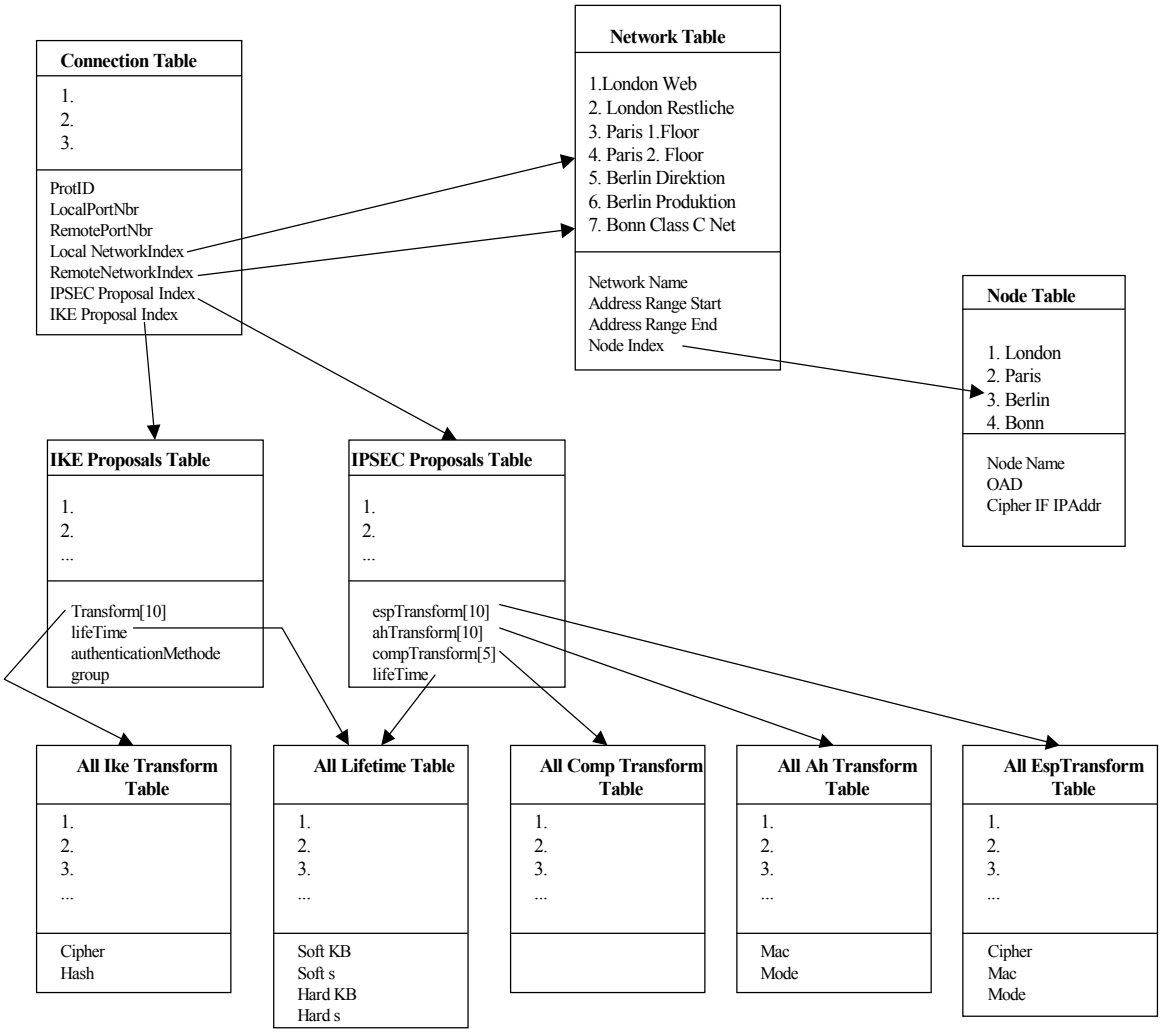
Wie schon im vorhergehenden Kapitel erwähnt, enthalten die Dateien je eine Tabelle. Die Tabellen dienen zur Speicherung der IPsec Konfigurationsparameter in der EDB (Embedded Database) der Omnisec 41x Produktfamilie. Diese Produktfamilie umfasst drei Geräte (Omnisec 410, 411 und 412), welche zur Verschlüsselung von Daten in einem VPN (Virtual Private Network) dienen. Mehr Informationen zu diesen Geräten findet man auf der Homepage von Omnisec: http://www.omnisec.ch

Es folgt eine Auflistung aller Datei-Namen und der entsprechenden Tabellen-Namen:

Datei	Tabelle
node.tab	Node Table
network.tab	Network Table
connection.tab	Connection Table
ikeproposal.tab	IKE Proposals Table
ipsecproposal.tab	IPSEC Proposals Table
iketransform.tab	All IKE Transform Table
esptransform.tab	All ESP Tranform Table
ahtransform.tab	All AH Transform Table
lifetime.tab	All Lifetime Table

Wie die Tabellen miteinander verknüpft sind sieht man im folgenden Table Model Diagramm.

Table Model



10.2.2.2 Konzept

Anforderungen
Eine wichtige Anforderung ist das schnell Laden der Daten. Daraus ergibt sich, dass die Tabellen so zur Verfügung gestellt werden, dass die Daten ohne die Verwendung von Suchfunktionen nur mittels Index-Zugriff gelesen werden können.

Indexing
Das hier verwendete Table Model bietet keine speziellen Datenbankfunktionen wie Primärschlüssel, automatische Checks (wenn Einträge gelöscht werden) etc. an.
Die Tabellen, welche in der EDB gespeichert werden, sind „direkt" indiziert. Das heisst, ein Index zeigt direkt auf den gewünschten Wert in der betreffenden Tabelle.
Beim Entfernen eines Elements aus der Node Table sind vielleicht nicht mehr alle Verweise der Network Table auf die Node Table gültig. Dies aus zwei Gründen:

1. Es existiert ein Verweis auf das Element, welches gelöscht wurde.
2. Durch das Entfernen eines Elements in der Node Table verkleinern sich die Indizes aller Einträge die nach dem gelöschten Element sind um 1. Dies erzwingt, dass die Einträge in der Network Table evtl. angepasst werden müssen.

10.2.2.3 Inhalt der Tabellen

Connection Table
Die Connection Table enthält alle die Informationen, welche Verbindungen auf welche Art geschützt werden. Standardmässig sind alle Verbindungen, welche nicht in dieser Tabelle vorkommen verboten.
Ein Connection Table Eintrag enthält folgende Informationen:

protID	32 Bit unsigned	Protokoll Identifier (z.B. UDP, TCP, ...)
localPortNbr	16 Bit unsigned	Local Port Number
remotePortNbr	16 Bit unsigned	Remote Port Number
ipsecProposal Index	unsigned char	Index auf Eintrag in der IPsec Proposal Tabelle
ikeProposalIndex	unsigned char	Index auf Eintrag in der IKE Proposal Tabelle
localNetwork Index	16 Bit unsigned	Index auf Network Table. Definiert lokales Netzwerk (Hinter diesem Security Gateway)
remoteNetwork Index	16 Bit unsigned	Index auf Network Table. Definiert Netzwerk welches hinter einem anderen Security Gateway liegt

Network Table
Die Network Table enthält alle Netzwerke, die dem Encryptor (= Security Gateway) bekannt sind. Bei diesen Netzwerken handelt es sich um Netze, welche hinter einem Security Gateway stehen (IP Adresse). Auch die Netze, welche hinter dem lokalen Security Gateway stehen sind in dieser Liste. Es wird nicht mit Netzwerkmasken und Subnetzen gearbeitet, da dies zu wenig flexibel ist.
Ein Network Management Center würde hier vielleicht alle Netze speichern, die im ganzen Netzwerk vorkommen. Dies ist kein Problem braucht aber mehr Speicher.
Ein Network Table Eintrag enthält folgende Informationen:

Network Name	Name des Netzwerkes 16 Byte lang	Netzwerknamen werden für IPsec nicht verwendet und können frei gewählt werden (Es empfiehlt sich Netzwerkweit die gleichen Namen für gleiche Netze zu verwenden).
RangeStart	IP Adresse 32 Bit unsigned	Erste IP Adresse des Netzwerkes z.B. 192.168.1.129 = C0A80181
RangeEnd	IP Adresse 32 Bit unsigned	Letzte IP Adresse des Netzwerkes Die letzte kann gleich der Ersten sein (Host)
nodeIndex	16 Bit unsigned	Index auf den Eintrag in der Node Tabelle, in welchem steht, hinter welcher IP-Adresse (Security Gateway) dieses Netzwerk liegt

Node Table

Die Node Table enthält alle Knoten welche der Encryptor (= Security Gateway) kennen muss. Ein Network Management Center würde hier vielleicht alle Knoten speichern, die im ganzen Netz vorkommen. Dies ist kein Problem braucht aber mehr Speicher.

Ein Node Table Eintrag enthält folgende Informationen:

OAD	Knotennummer 32 Bit unsigned	Die OAD muss eineindeutig sein, d.h. zwei Knoten dürfen nicht dieselbe Nummer besitzen.
Node Name	Name des Knotens 16 Byte lang	Nodenames werden für IPsec nicht verwendet und können frei gewählt werden (Es empfiehlt sich Netzwerkweit die gleichen Namen für gleiche Knoten zu verwenden).
Cipher IF IP Addr	32 Bit unsigned	IP Adresse des Cipher Interface

IKE Proposal Table

Eine Proposal Table enthält die Information, welche Proposals das Gerät beim Keymanagement für die IPsec Session anbieten soll.

Ein Proposal Table Eintrag enthält folgende Informationen:

IKETransform	Array mit 10 Elementen vom Typ unsigned char	Liste mit IKE Transforms, welche erlaubt sind.
Lifetime	unsigned char	Index auf die AllLifeTimeTable
AuthenticationMethode	unsigned char	
Group	unsigned char	

IKE Transform
Im IKE Transform ist angegeben welche Algorithmen erlaubt sind. Es können maximal 10 Transform selektiert werden. Zum Beispiel bedeutet
IKETransform[0] = 2;
IKETransform[1] = 4;
alle anderen = 0

dass folgende Varianten bei IKE zugelassen sind

Vorschlag Nr	Chiffrierung	Hash
1	IkeIdeaCbc	IkeMd5
2	IkeRc5R16B64Cbc	IkeMd5

Es ist ersichtlich, dass die Zahlen in IKE Transform direkt den Index in „allIkeTransformTable“ enthält

```
static const IkeTransform allIkeTransformTable[] =
{
    { IkeNoCipher,     IkeNoHash },
    { IkeDesCbc,       IkeMd5 },
    { IkeIdeaCbc,      IkeMd5 },
    { IkeBlowfishCbc,  IkeMd5 },
    { IkeRc5R16B64Cbc, IkeMd5 },
    { IkeDes3Cbc,      IkeMd5 },
...
}
```

LifeTime
Die LifeTime ist ein Index auf die allLifeTimeTable, welche wie folgt definiert ist.

```
static const SshIpmSaLifetime allLifetimeTable[] =
{
    { 0, 0, 0, 0},
    { 0, 0, 1000, 3600 },
};
```

Authentication Methode
Die Authentication Methode ist wie folgt definiert

```
typedef enum
{
  SSH_IPM_IKE_AUTH_METH_PRE_SHARED_KEY          = 1,
  SSH_IPM_IKE_AUTH_METH_DSS_SIGNATURES          = 2,
  SSH_IPM_IKE_AUTH_METH_RSA_SIGNATURES          = 3,
  SSH_IPM_IKE_AUTH_METH_RSA_ENCRYPTION          = 4,
  SSH_IPM_IKE_AUTH_METH_RSA_ENCRYPTION_REVISED  = 5
} SshIpmIkeAuthMethod;
```

Group
Für Diffie-Hellman key Exchange (nur benützt bei public key Exchange) Muss vorläufig 0 sein (public key exchange wird noch nicht unterstützt).

IPsec Proposal Table
Eine Proposal Table enthält die Information, welche Proposals das Gerät beim Keymanagement für die IPsec Session anbieten soll.
Ein Proposal Table Eintrag enthält folgende Informationen:

EspTransform	Array mit 10 Elementen vom Typ unsigned char	Liste mit ESP Transforms, welche erlaubt sind.
AhTransform	Array mit 10 Elementen vom Typ unsigned char	Liste mit AH Transforms, welche erlaubt sind.
CompTransform	Array mit 5 Elementen vom Typ unsigned char	Liste mit Compression Transforms, welche erlaubt sind.
Lifetime	unsigned char	

Esp Transform
Im Esp Transform ist angegeben welche Algorithmen erlaubt sind. Es können maximal 10 Transforms selektiert werden. Zum Beispiel bedeutet
EspTransform[0] = 2;
EspTransform[1] = 5;
alle anderen = 0

dass folgende Varianten bei IPsec zugelassen sind:

Vorschlag Nr	Chiffrierung	Mac	Mode
1	IpsecDesIv64	IpsecMd5	Transport
2	IpsecDes3	IpsecMd5	Tunnel

Es ist ersichtlich, dass die Zahlen in Esp Transform direkt den Index in „allEspTransformTable“ enthält

```
static const EspTransform allEspTransformTable [] =
{
    { IpsecNoCipher, IpsecNoMac, NoMode },
    { IpsecDesIv64,  IpsecMd5,   Tunnel },
    { IpsecDesIv64,  IpsecMd5,   Transport },
    { IpsecDes,      IpsecMd5,   Tunnel },
    { IpsecDes,      IpsecMd5,   Transport },
    { IpsecDes3,     IpsecMd5,   Tunnel },
    { IpsecDes3,     IpsecMd5,   Transport },
...
};
```

Ah Transform
Im Ah Transform ist angegeben welche Algorithmen erlaubt sind. Es können maximal 10 Transforms selektiert werden. Zum Beispiel bedeutet
AhTransform[0] = 1;
AhTransform[1] = 3;
alle anderen = 0

dass folgende Varianten bei IKE zugelassen sind

Vorschlag Nr	Authentisierung	Mode
1	IpsecMd5	Tunnel
2	IpsecSha	Tunnel

Es ist ersichtlich, dass die Zahlen in IKE Transform direkt den Index in „allIkeTransformTable“ enthält

```
static const AhTransform allAhTransformTable [] =
{
    { IpsecNoMac,  NoMode },
    { IpsecMd5,    Tunnel },
    { IpsecMd5,    Transport },
    { IpsecSha,    Tunnel },
    { IpsecSha,    Transport },
    { IpsecMacDes, Tunnel },
    { IpsecMacDes, Transport },
...
};
```

Comp Transform
Im Comp Transform ist angegeben welche Kompressions Algorithmen erlaubt sind. Es können maximal 5 Transforms selektiert werden. Zum Beispiel bedeutet
CompTransform[0] = 1;
CompTransform[1] = 2;
alle anderen = 0

dass folgende Algorithmen bei der Kompression zugelassen sind

Vorschlag Nr	Algorithmus
1	IpsecOui
2	IpsecDeflate

Es ist ersichtlich, dass die Zahlen in Comp Transform direkt den Index in „allCompTransformTable“ enthält

```
{ static const CompTransform allCompTransformTable [] =
{
    { IpsecNoCompression },
    { IpsecOui },
    { IpsecDeflate },
    { IpsecLzs },
};
```

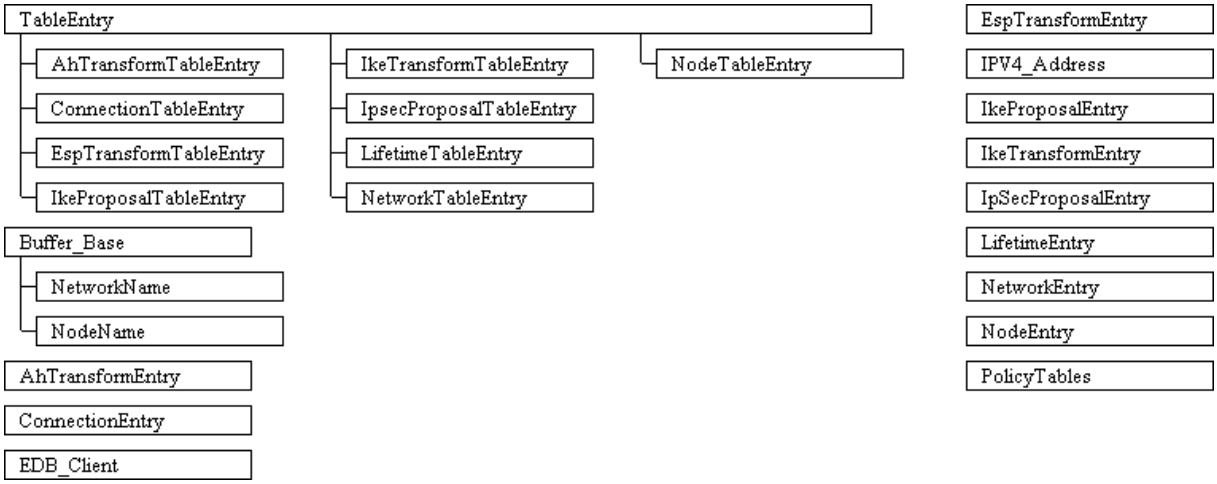
LifeTime
Die LifeTime ist ein Index auf die allLifeTimeTable, welche wie folgt definiert ist.

```
static const SshIpmSaLifetime allLifetimeTable[] =
{
    { 0, 0, 0, 0},
    { 0, 0, 1000, 3600 },
};
```

10.2.2.4 Quellcode-Dateien

Dateien	Enthaltene Klassen in Datei	Beschreibung
basictypes.hpp	-	Spezielle Grundtypen (z.B. UInt16, ...)
oagtypes.hpp, oagtypes.cpp	IPV4_Address NodeName NetworkName NodeEntry NetworkEntry ConnectionEntry IpSecProposalEntry IkeProposalEntry IkeTransformEntry EspTransformEntry AhTransformEntry LifetimeEntry	Omnisecspezifische Datentypen
oagtable.hpp, oagtable.cpp	NodeTableEntry NetworkTableEntry ConnectionTableEntry IpsecProposalTableEntry IkeProposalTableEntry IkeTransformTableEntry EspTransformTableEntry AhTransformTableEntry LifetimeTableEntry	Omnisec-spezifische Tabellen-Datensätze
oag_util.hpp, oag_util.cpp	Buffer_Base	Omnisec-spezifische Werkzeuge (Definition eines Puffers, der zur Zwischenspeicherung der Tabellen-Datensätze dient)
edb_tables.hpp, edb_tables.cpp	TableEntry EDB_Client PolicyTables	EDB-Schnittstelle (EDB = Embedded Database)
main.cpp	-	Hauptprogramm

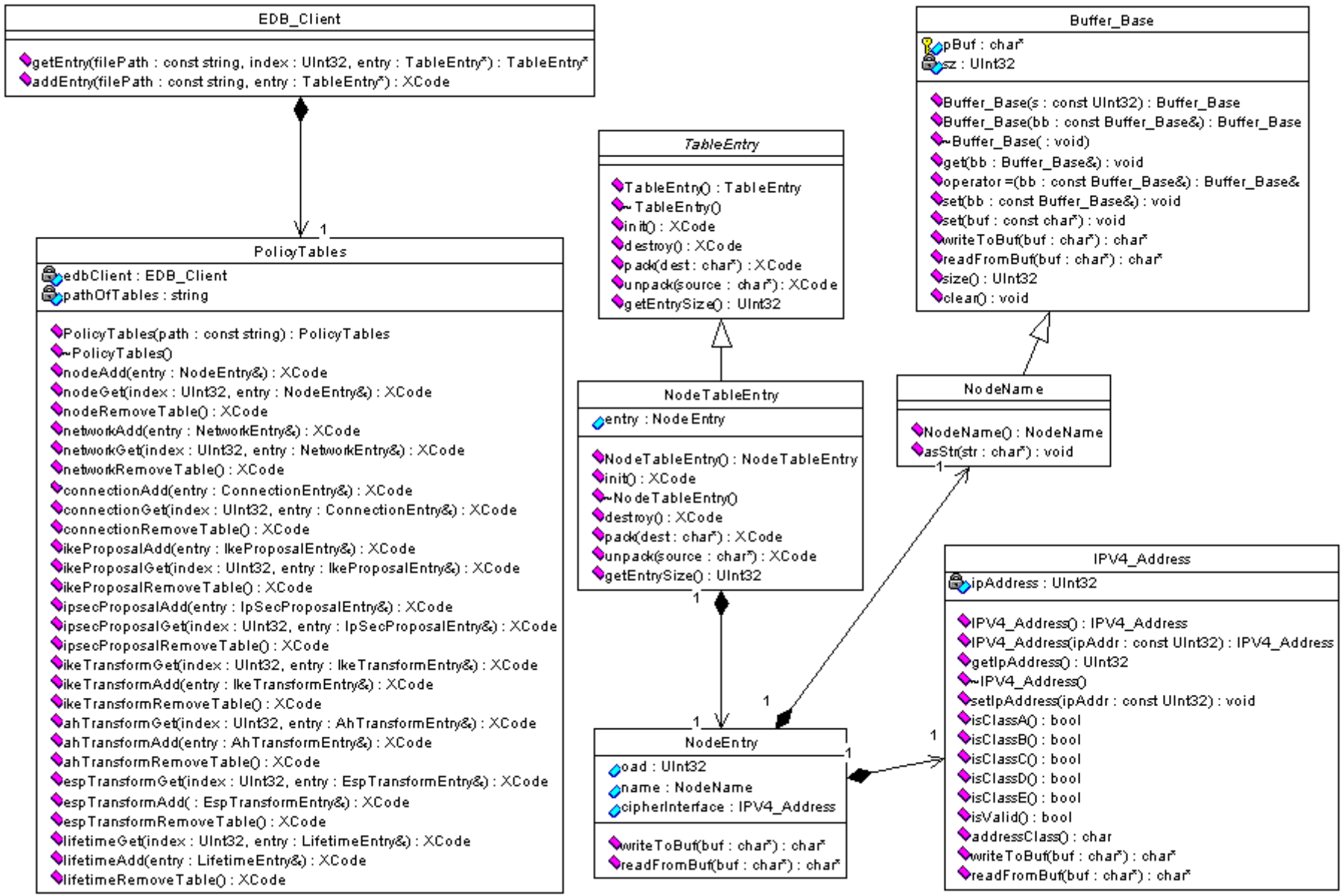
10.2.2.5 Klassendiagramm (Gesamtübersicht aller 25 Klassen)



Im obigen Klassendiagramm sind alle Klassen vorhanden, die in "IpTables" enthalten sind.

Folgende Klassen funktionieren nach dem gleichen Prinzip, da sie von der gleichen Klasse abgeleitet werden und da jede für die Umsetzung der 9 Tabellen in eine Datei verantwortlich sind: AhTransformTableEntry, ConnectionTableEntry, EspTransformTableEntry, IkeProposalTableEntry, IkeTransformTableEntry, IpsecProposalTableEntry, LifetimeTableEntry, NetworkTableEntry und NodeTableEntry (alle von **TableEntry** abgeleitet) ("Gruppe 1").
Dasselbe gilt für die Klassen NetworkName und NodeName (alle von **Buffer_Base** abgeleitet) ("Gruppe 2").
Die nächsten 9 Klassen sind nicht abgeleitete Klassen: AhTransformEntry, ConnectionEntry, EspTransformEntry, IkeProposalEntry, IkeTransformEntry, IpsecProposalEntry, LifetimeEntry, NetworkEntry und NodeEntry ("Gruppe 3").

Im folgenden Klassendiagramm sind nicht mehr alle Klassen enthalten. Stellvertretend für eine Gruppe haben wir jeweils nur eine Klasse ausgewählt: **NodeTableEntry**, **NodeName** und **NodeEntry**. Dieses Diagramm gilt also sinngemäss auch für alle anderen Klassen aus den Gruppen 1 bis 3 (z.B. NetworkTableEntry, NetworkName und NetworkEntry). So lässt sich das Design wesentlich leichter erklären und darstellen (nur noch 8 Klassen zu erklären).


EDB_Client
getEntry(filePath : const string, index : UInt32, entry : TableEntry*) : TableEntry*
addEntry(filePath : const string, entry : TableEntry*) : XCode
1
PolicyTables
edbClient : EDB_Client
pathOfTables : string
PolicyTables(path : const string) : PolicyTables
~PolicyTables()
nodeAdd(entry : NodeEntry&) : XCode
nodeGet(index : UInt32, entry : NodeEntry&) : XCode
nodeRemoveTable() : XCode
networkAdd(entry : NetworkEntry&) : XCode
networkGet(index : UInt32, entry : NetworkEntry&) : XCode
networkRemoveTable() : XCode
connectionAdd(entry : ConnectionEntry&) : XCode
connectionGet(index : UInt32, entry : ConnectionEntry&) : XCode
connectionRemoveTable() : XCode
ikeProposalAdd(entry : IkeProposalEntry&) : XCode
ikeProposalGet(index : UInt32, entry : IkeProposalEntry&) : XCode
ikeProposalRemoveTable() : XCode
ipsecProposalAdd(entry : IpSecProposalEntry&) : XCode
ipsecProposalGet(index : UInt32, entry : IpSecProposalEntry&) : XCode
ipsecProposalRemoveTable() : XCode
ikeTransformGet(index : UInt32, entry : IkeTransformEntry&) : XCode
ikeTransformAdd(entry : IkeTransformEntry&) : XCode
ikeTransformRemoveTable() : XCode
ahTransformGet(index : UInt32, entry : AhTransformEntry&) : XCode
ahTransformAdd(entry : AhTransformEntry&) : XCode
ahTransformRemoveTable() : XCode
espTransformGet(index : UInt32, entry : EspTransformEntry&) : XCode
espTransformAdd(: EspTransformEntry&) : XCode
espTransformRemoveTable() : XCode
lifetimeGet(index : UInt32, entry : LifetimeEntry&) : XCode
lifetimeAdd(entry : LifetimeEntry&) : XCode
lifetimeRemoveTable() : XCode
TableEntry
TableEntry() : TableEntry
~TableEntry()
init() : XCode
destroy() : XCode
pack(dest : char*) : XCode
unpack(source : char*) : XCode
getEntrySize() : UInt32
NodeTableEntry
entry : NodeEntry
NodeTableEntry() : NodeTableEntry
init() : XCode
~NodeTableEntry()
destroy() : XCode
pack(dest : char*) : XCode
unpack(source : char*) : XCode
getEntrySize() : UInt32
1
1
NodeEntry
oad : UInt32
name : NodeName
cipherInterface : IPV4_Address
writeToBuf(buf : char*) : char*
readFromBuf(buf : char*) : char*
Buffer_Base
pBuf : char*
sz : UInt32
Buffer_Base(s : const UInt32) : Buffer_Base
Buffer_Base(bb : const Buffer_Base&) : Buffer_Base
~Buffer_Base(: void)
get(bb : Buffer_Base&) : void
operator =(bb : const Buffer_Base&) : Buffer_Base&
set(bb : const Buffer_Base&) : void
set(buf : const char*) : void
writeToBuf(buf : char*) : char*
readFromBuf(buf : char*) : char*
size() : UInt32
clear() : void
NodeName
NodeName() : NodeName
asStr(str : char*) : void
1
1
1
1
IPV4_Address
ipAddress : UInt32
IPV4_Address() : IPV4_Address
IPV4_Address(ipAddr : const UInt32) : IPV4_Address
getIpAddress() : UInt32
~IPV4_Address()
setIpAddress(ipAddr : const UInt32) : void
isClassA() : bool
isClassB() : bool
isClassC() : bool
isClassD() : bool
isClassE() : bool
isValid() : bool
addressClass() : char
writeToBuf(buf : char*) : char*
readFromBuf(buf : char*) : char*

NodeTableEntry ist von der abstrakten Basisklasse TableEntry abgeleitet und enthält ein Klassenelement (Attribut) namens entry, das vom Klassentyp NodeEntry ist. Die Klasse NodeEntry enthält die Klassenelemente oad, name (Klassentyp NodeName) und cipherInterface (Klassentyp IPV4_Address). Schliesslich ist NodeName noch von Buffer_Base abgeleitet.

PolicyTables enthält ein Element der Klasse EDB_Client.

Klasse	Erklärung
EDB_Client	Schnittstelle zu den Tabellen
PolicyTables	Dient zum Abfüllen bzw. Abfragen einzelner Werte in der entsprechenden Tabelle. Zusätzlich können ganze Tabellen (von der Harddisk) gelöscht werden.
NodeEntry	Datensatz in der Tabelle Node Table
NodeTableEntry	Dient zur Verwaltung der Node Table Datensätze
TableEntry	Ein Pointer der abstrakten Basisklasse TableEntry kann auch Objekte der abgeleiteten Klassen referenzieren. Dieser Designentscheid ermöglicht es, dass die Methoden der Klasse EDB_Client unterschiedliche Objekte (d.h. Objekte der Klassen NodeTableEntry, NetworkTableEntry, ConnectionTableEntry, ...) verwalten können.
Buffer_Base	Dient zur Verwaltung eines Puffers.
NodeName	Puffer für einen Node Name
IP4_Address	Dient zur Verwaltung von IP4-Addressen.

10.2.2.6 Erklärungen zu einigen wichtigen Methoden:

Methoden	Dateien	Klassen	Erklärung
writeToBuf, readFromBuf	▪ oagtypes.hpp ▪ oagtypes.cpp ▪ oag_util.hpp ▪ oag_util.cpp	▪ IPV4_Address ▪ NodeEntry ▪ NetworkEntry ▪ ConnectionEntry ▪ IpSecProposalEntry ▪ IkeProposalEntry ▪ IkeTransformEntry ▪ EspTransformEntry ▪ AhTransformEntry ▪ LifetimeEntry	Diese Methoden schreiben bzw. lesen in bzw. von einen Puffer, der durch einen char-Pointer referenziert wird.
pack, unpack	▪ oagtable.hpp ▪ oagtable.cpp ▪ edb_tables.hpp ▪ edb_tables.cpp	▪ NodeTableEntry ▪ NetworkTableEntry ▪ ConnectionTableEntry ▪ IpsecProposalTableEntry ▪ IkeProposalTableEntry ▪ IkeTransformTableEntry ▪ EspTransformTableEntry ▪ AhTransformTableEntry ▪ LifetimeTableEntry ▪ TableEntry	Die Methode pack ruft im wesentlichen nur die Methode writeToBuf des Klassenelements entry auf. Die Methode unpack ruft im wesentlichen nur die Methode readFromBuf des Klassenelements entry auf.
getEntrySize	▪ oagtable.hpp ▪ oagtable.cpp ▪ edb_tables.hpp ▪ edb_tables.cpp	▪ NodeTableEntry ▪ NetworkTableEntry ▪ ConnectionTableEntry ▪ IpsecProposalTableEntry ▪ IkeProposalTableEntry ▪ IkeTransformTableEntry ▪ EspTransformTableEntry ▪ AhTransformTableEntry ▪ LifetimeTableEntry ▪ TableEntry	Diese Methode berechnet die Grösse des entsprechenden "TableEntries" (z.B. bei der Klasse NodeTableEntry wird die Grösse des Objektes NodeEntry zurückgegeben).
getEntry	▪ edb_tables.hpp ▪ edb_tables.cpp	▪ EDB_Client	Liest einen Tabellen-Datensatz von der Harddisk.
addEntry	▪ edb_tables.hpp ▪ edb_tables.cpp	▪ EDB_Client	Schreibt einen Tabellen-Datensatz auf die Harddisk.

Für weitere Informationen über das Verhalten bestimmter Methoden vergleiche man bitte den Quellcode der gut kommentiert ist.

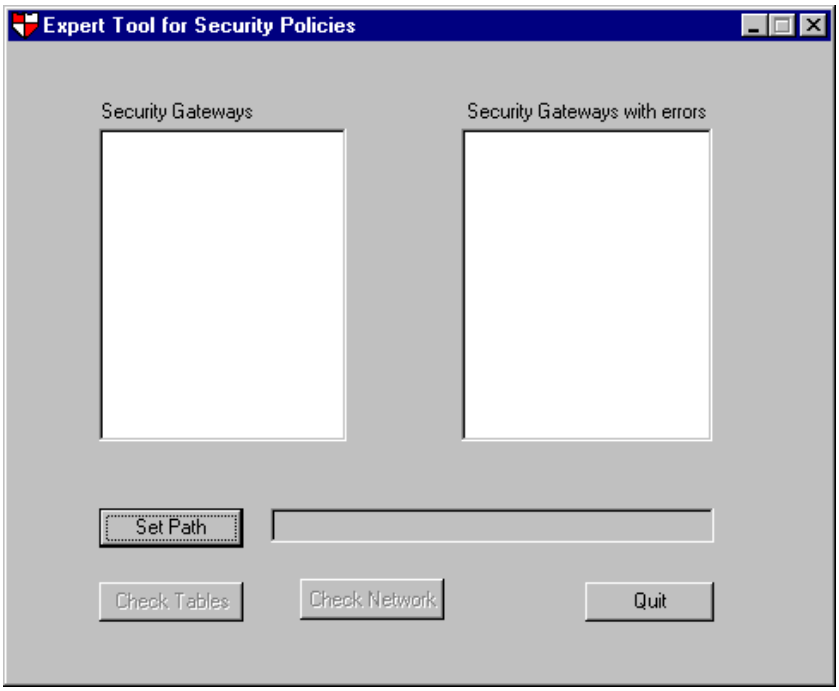
10.3 Programm "Expert Tool for Security Policies"

10.3.1 Benutzeranleitung

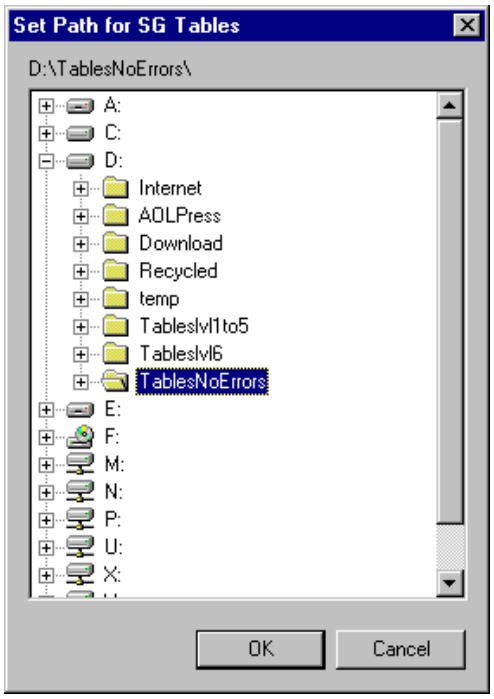
Beim ausführen des Programmes gibt es drei verschiedene Verhaltensmöglichkeiten. Zum einen kann das Programm fehlerfrei ausgeführt werden (d.h. es werden keine Fehler in den Tabellen entdeckt). Falls das Programm jedoch in den Tabellen Fehler entdeckt, verhält es sich ein wenig anders. Die Tests werden in 2 Schritten ausgeführt. Falls der erst Test nicht fehlerfrei abgeschlossen wird, kann der zweite Test nicht gestartet werden.

10.3.1.1 Fehlerfrei

Nachdem Sie das Programm gestartet haben (ExpertTool.exe) erscheint folgendes Fenster.



Klicken Sie nun den Knopf „Set Path“ um das Arbeitsverzeichnis zu wählen.

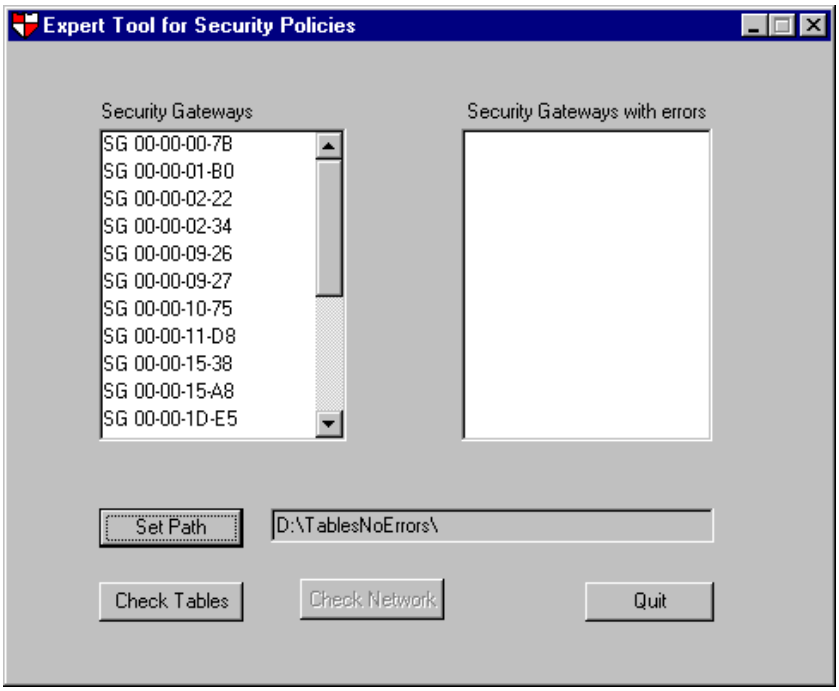


Es erscheint ein neues Fenster, welches dem auf der linken Seite ähnlich sieht.

Falls Sie im Laufwerk „D:“ ein Verzeichnis namens „Tables“ haben, wird dieses automatisch angewählt.

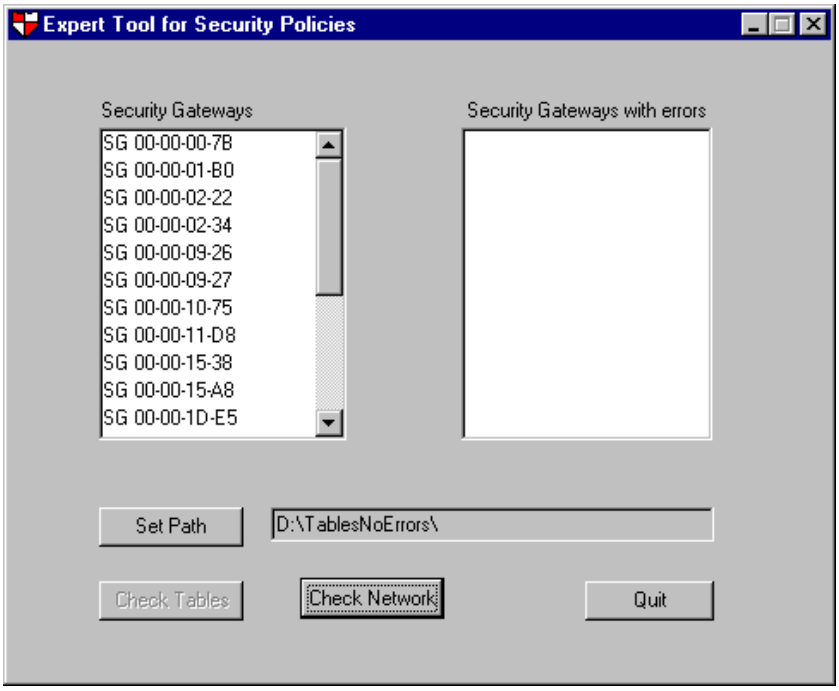
Falls nicht, wählen Sie nun das gewünschte Verzeichnis aus und klicken Sie auf „OK“

Es erscheint nun wieder das erste Fenster mit ein paar kleinen Unterschieden. Im Fenster „Security Gateways“ sind nun alle vorhandenen Security Gateways aufgeführt. Rechts neben dem Knopf „Set Path“ erscheint das aktuelle Verzeichnis und der Knopf „Check Tables“ ist nun aktiviert.



Beim Klicken auf den Knopf „Check Tables“ wir die erste Testserie gestartet. Dies kann einige Minuten dauern. Es erscheint ein Fenster, auf welchem man den Fortschritt verfolgen kann.

(1) Falls es bei den Tests keinen Fehler gab, sollten Sie nun folgendes Fenster vor sich haben.



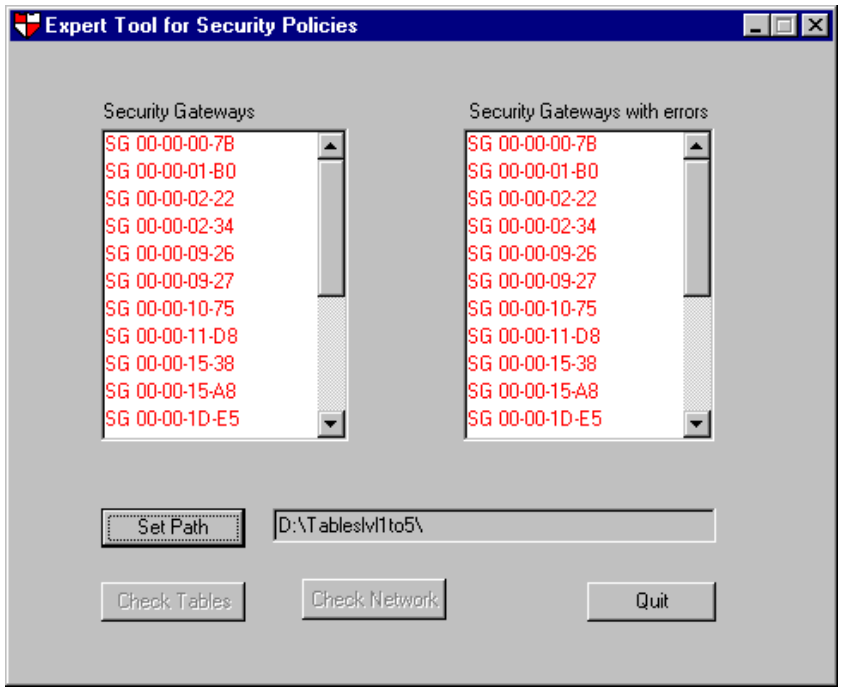
(2) Der Unterschied zum vorherigen Fenster besteht darin, dass der Knopf „Check Tables“ nicht mehr aktiv ist, dafür der Knopf „Check Network“. Mit diesem Knopf kann man nun die zweite und letzte Testreihe einleiten.

Es erscheint wieder das Fenster, welches den Fortschritt anzeigt. Falls dieser Test auch keine Fehler gefunden hat, wird das nebenstehende Fenster erscheinen

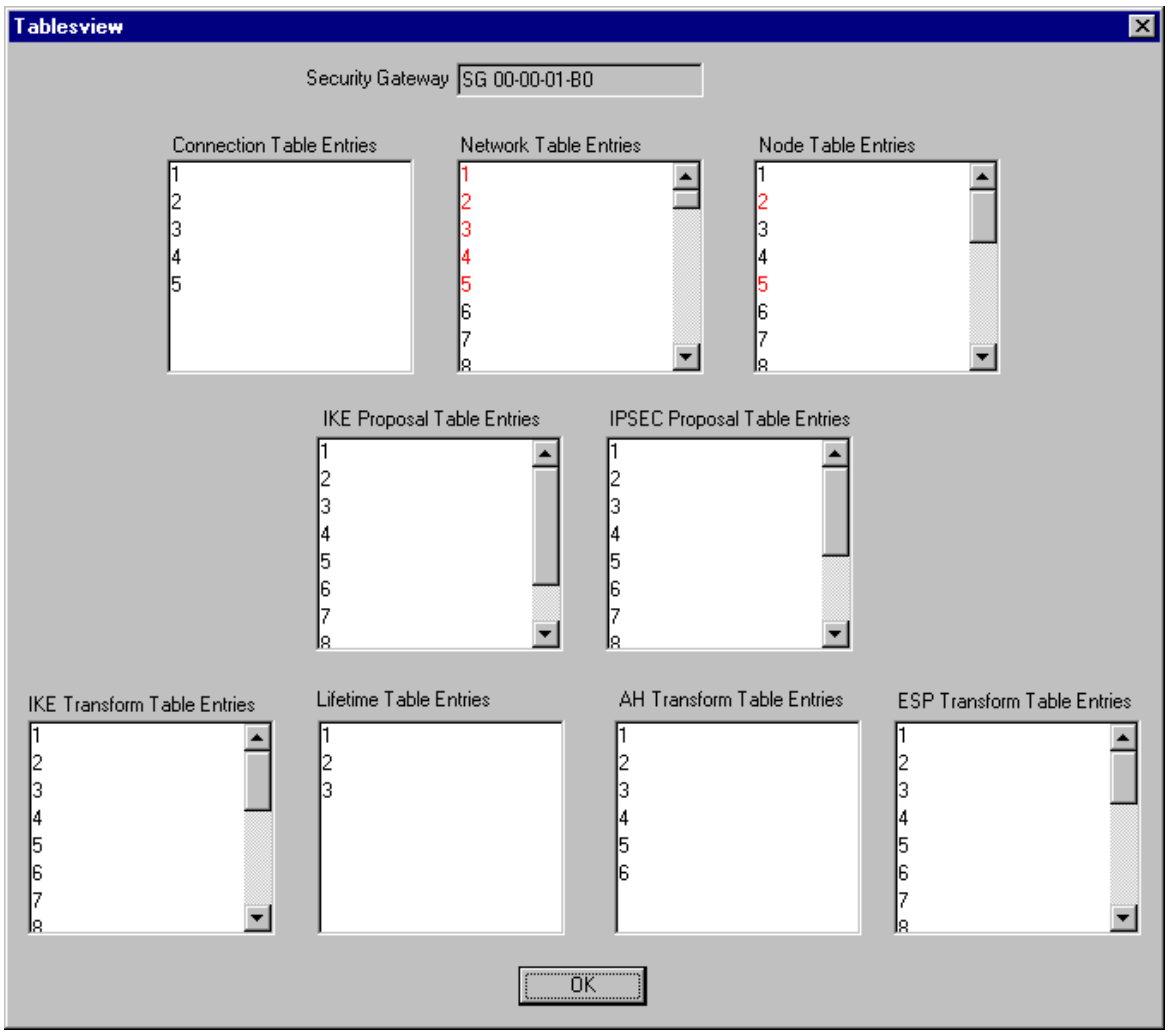
Sie können nun das Programm mit „Quit“ beenden oder mit „Set Path“ ein neues Verzeichnis auswählen.

10.3.1.2 Fehler beim ersten Test

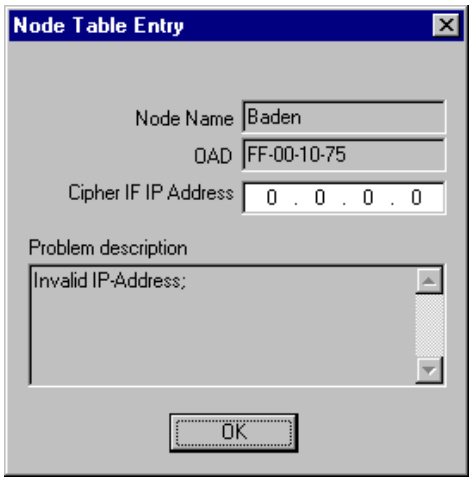
Falls es bei (1) einen Fehler gab, erscheint folgendes Fenster.

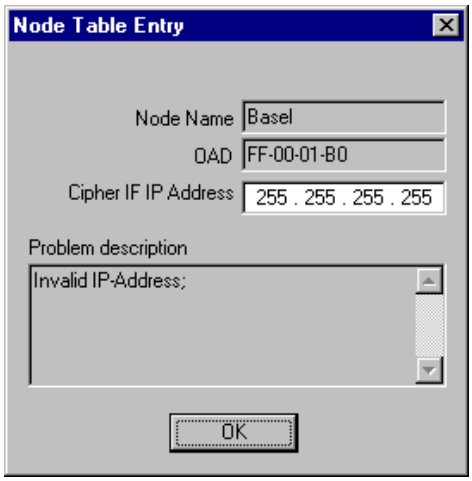


Im linken Fenster werden all Security Gateways mit einem Fehler rot markiert und im rechten Fenster werden diese rot aufgelistet. Falls Sie nun im rechten Fenster mit einem Doppelklick ein Security Gateway auswählen, erscheint eine Übersicht für alle Tabellen. Man beachte, dass die Knöpfe „Check Tables“ und „Check Network“ beide nicht aktiv sind, da der nächste Test fehlerfreie Tabellen benötigt.

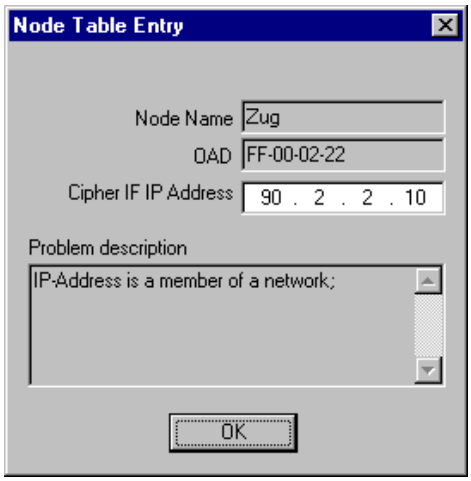


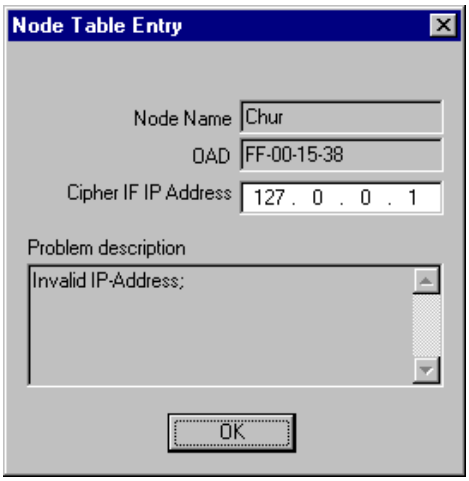
In den verschiedenen Fenstern sind alle Einträge der entsprechenden Tabelle aufgelistet. Fehlerhafte Einträge sind rot gekennzeichnet. Mit einem Doppelklick auf einen Eintrag kann man die Detailinformationen einsehen. Nachfolgen sehen Sie einige Beispiele.

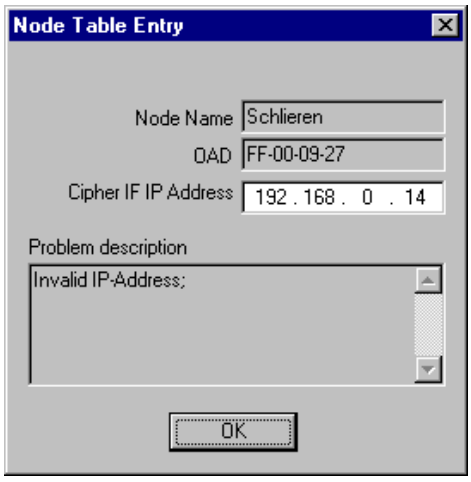


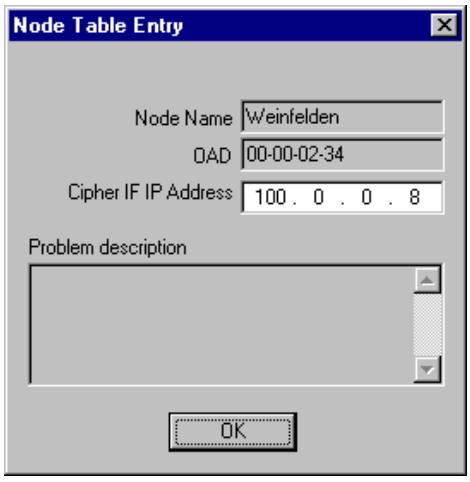


Bei den oberen beiden Fenstern erkennt man, dass bei beiden die IP-Adresse ungültig ist. Bei den unteren beiden ist im rechten die IP-Adresse ebenfalls falsch. Beim linken ist die IP-Adresse durch ein Netzwerk verwendet.

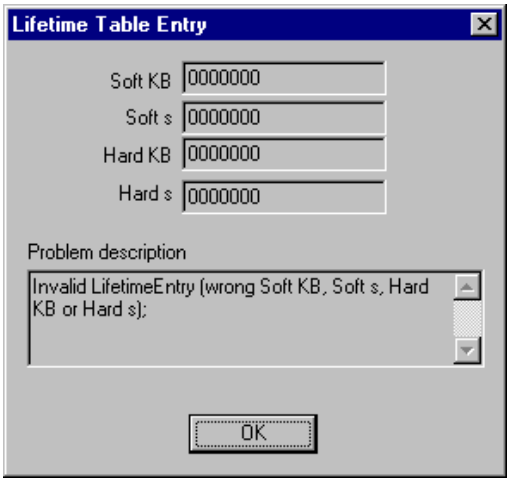


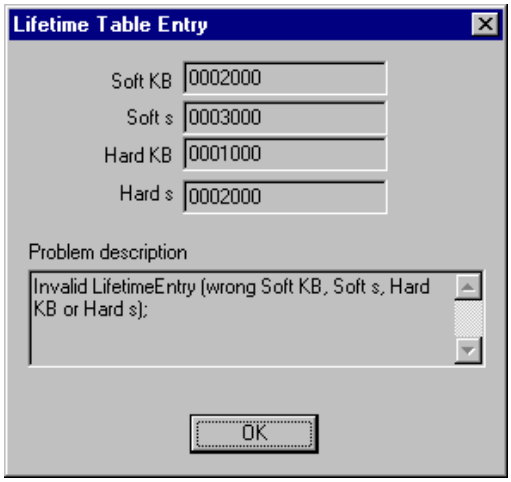


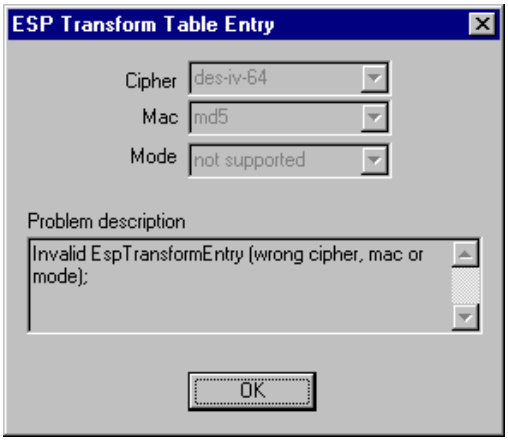




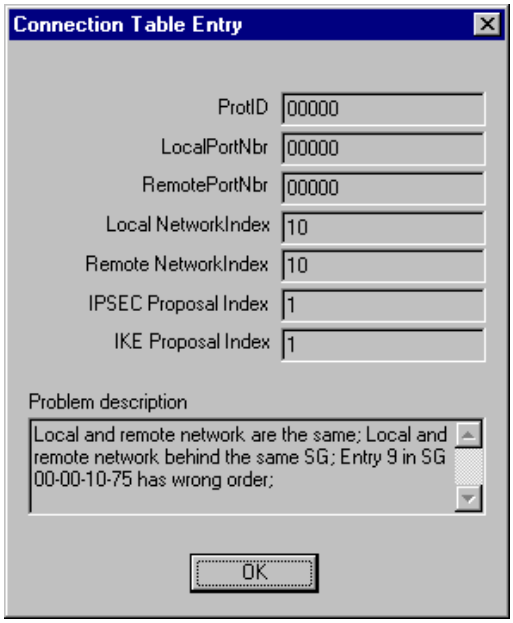
Die oberen Fenster sind auch wieder „NodeEntries“, das linke wieder mit einer ungültigen IP-Adresse und das rechte mit einem Eintrag, welcher in Ordnung ist. Unten sind zwei ungültige „LifetimeEntries“ dargestellt

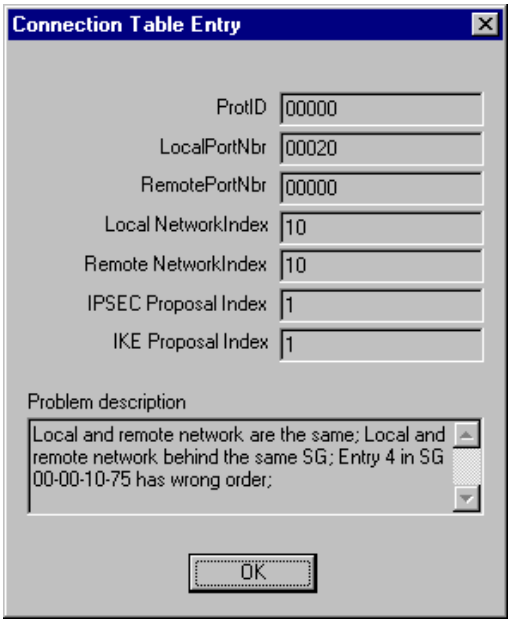






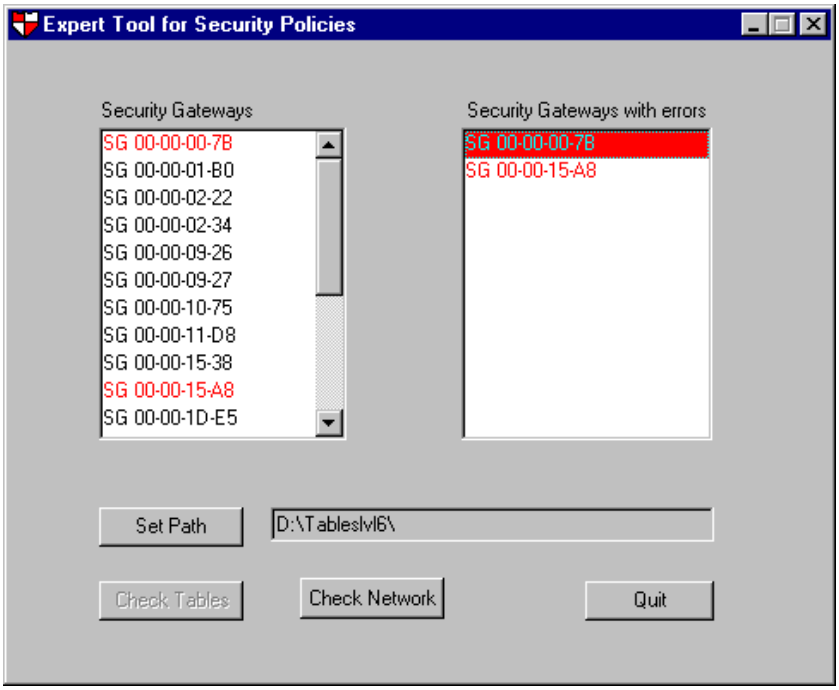
Unten sieht man nun schön, das bei bestimmten Fehlern auch der Eintrag erwähnt wird, mit dem das Problem besteht (links ist der Eintrag 4 und rechts Nummer 9)





10.3.1.3 Fehler nach dem zweiten Test

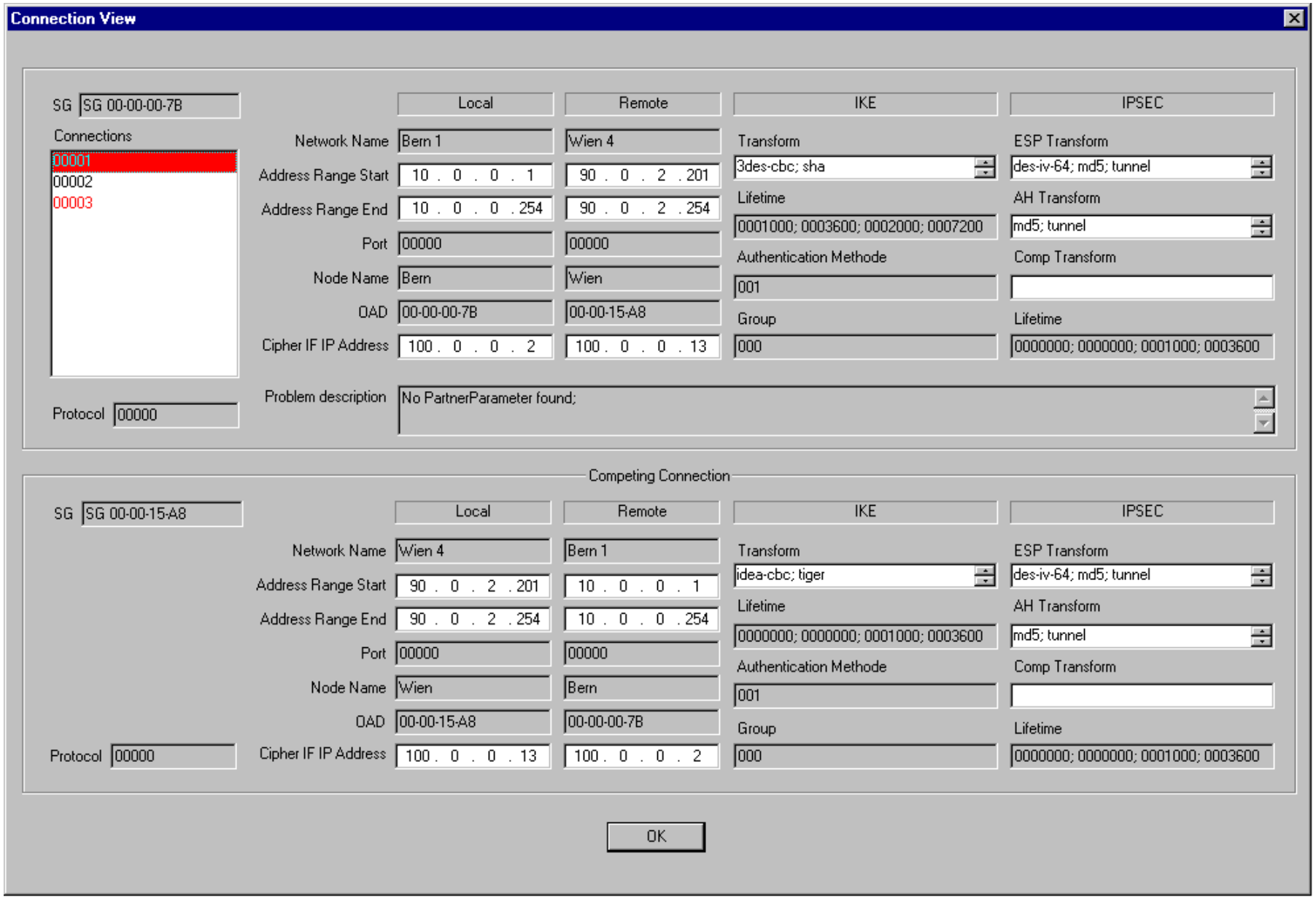
Falls bei (2) ein Fehler aufgetreten ist, kommt anstelle der „alles OK“ Meldung folgendes Fenster.



Alle Security Gateways mit Fehlerhaften Tabellen werden wiederum rot eingefärbt und im rechten Fenster aufgelistet. Ein Doppelklick auf einen Security Gateway auf der rechten Seite lässt das Fenster auf der folgenden Seite erscheinen. In der linken oberen Ecke sind alle Verbindungen dieses Security Gateways aufgelistet. Die Fehlerhaften sind rot markiert. Bei einem Klick auf die Nummer werden die Felder aufgefüllt. Es kann zwei verschiedene Fehler geben:

- Keine gemeinsamen Parameter
- Keine Partnerverbindung vorhanden

Beim ersten Fehler wird zusätzlich die untere Hälfte mit den Daten der Partner Verbindung gefüllt, was ein einfaches Vergleichen der beiden Verbindungen Ermöglicht.


Connection View
SG
SG 00-00-00-7B
Connections
00001
00002
00003
Protocol
00000
Local
Remote
IKE
IPSEC
Network Name
Bern 1
Wien 4
Address Range Start
10 . 0 . 0 . 1
90 . 0 . 2 . 201
Address Range End
10 . 0 . 0 . 254
90 . 0 . 2 . 254
Port
00000
00000
Node Name
Bern
Wien
OAD
00-00-00-7B
00-00-15-A8
Cipher IF IP Address
100 . 0 . 0 . 2
100 . 0 . 0 . 13
Transform
3des-cbc; sha
ESP Transform
des-iv-64; md5; tunnel
Lifetime
0001000; 0003600; 0002000; 0007200
AH Transform
md5; tunnel
Authentication Methode
001
Comp Transform
Group
000
Lifetime
0000000; 0000000; 0001000; 0003600
Problem description
No PartnerParameter found;
Competing Connection
SG
SG 00-00-15-A8
Local
Remote
IKE
IPSEC
Network Name
Wien 4
Bern 1
Address Range Start
90 . 0 . 2 . 201
10 . 0 . 0 . 1
Address Range End
90 . 0 . 2 . 254
10 . 0 . 0 . 254
Port
00000
00000
Node Name
Wien
Bern
OAD
00-00-15-A8
00-00-00-7B
Cipher IF IP Address
100 . 0 . 0 . 13
100 . 0 . 0 . 2
Protocol
00000
Transform
idea-cbc; tiger
ESP Transform
des-iv-64; md5; tunnel
Lifetime
0000000; 0000000; 0001000; 0003600
AH Transform
md5; tunnel
Authentication Methode
001
Comp Transform
Group
000
Lifetime
0000000; 0000000; 0001000; 0003600
OK

10.3.2 Softwaredesign

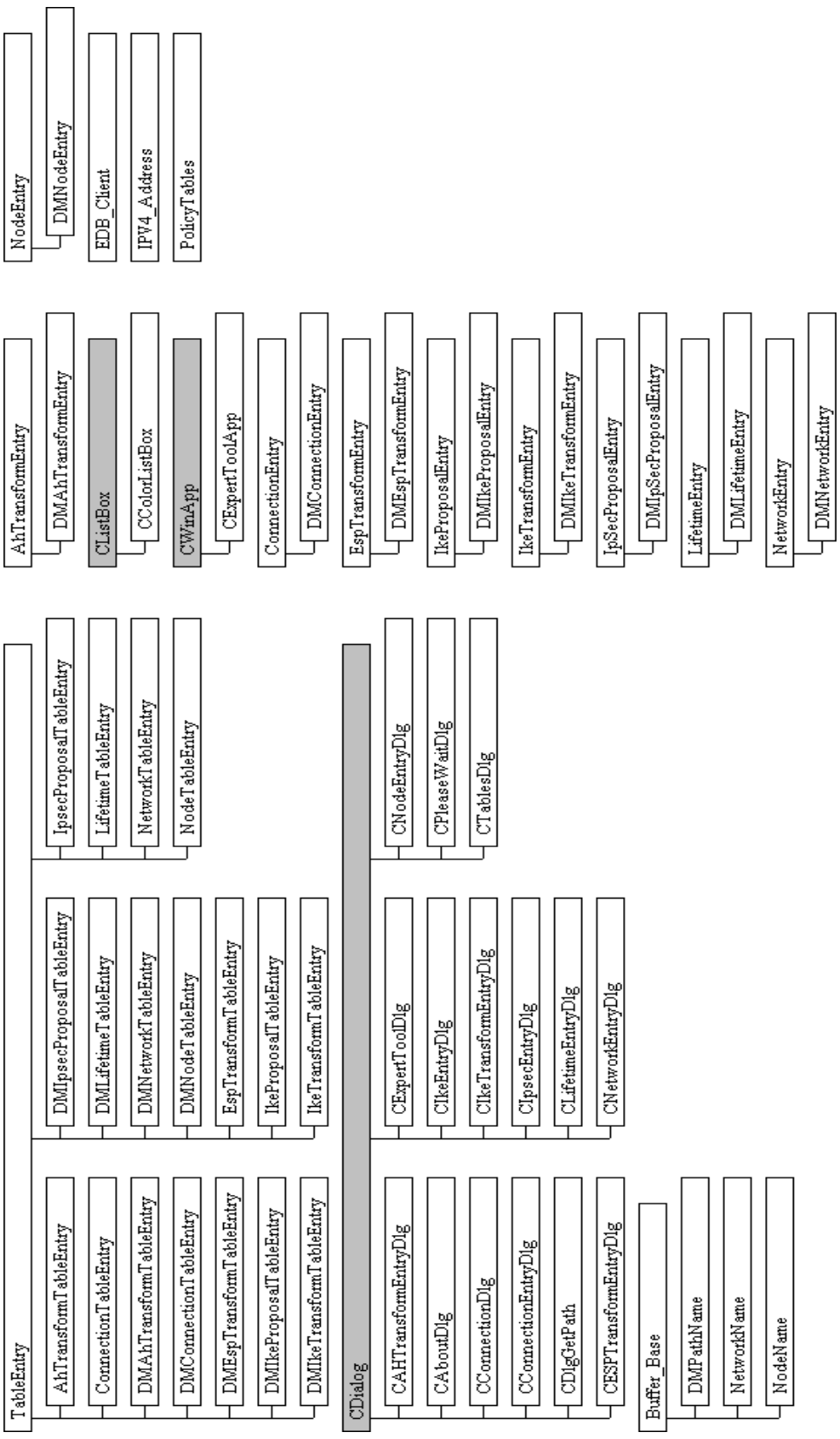
Wie schon im Kapitel 10.1 erwähnt ist unser Programm ExpertTool eine Erweiterung des Programmes IpTables (von Industriepartner Omnisec AG). Wir empfehlen unbedingt zuerst das Kapitel 10.2 zu lesen und erst dann hier weiterzumachen, da wir im Kapitel 10.2 erklären wie IpTables funktioniert. Falls man IpTables versteht, dann braucht es nicht mehr viel um auch ExpertTool zu verstehen.

10.3.2.1 Quellcode-Dateien

Dateien	Enthaltene Klassen in Datei	Beschreibung
basictypes.hpp, oagtypes.hpp, oagtypes.cpp, oagtable.hpp, oagtable.cpp, oag_util.hpp, oag_util.cpp, edb_tables.hpp, edb_tables.cpp	IPV4_Address NodeName NetworkName ...	Vom Programm IpTables übernommene Quellcode-Dateien
ExpertTool.h, ExpertTool.cpp	nicht nennenswerte MFC Klassen	Hauptprogramm
ExpertToolDlg.cpp ExpertToolDlg.h	nicht nennenswerte MFC Klassen	Hauptdialog
tablesdlg.cpp tablesdlg.h	nicht nennenswerte MFC Klassen	Tabellen-Übersichtsdialog
AHTransformEntryDlg.cpp AHTransformEntryDlg.h ConnectionEntryDlg.cpp ConnectionEntryDlg.h ESPTransformEntryDlg.cpp ESPTransformEntryDlg.h IkeEntryDlg.cpp IkeEntryDlg.h IkeTransformEntryDlg.cpp IkeTransformEntryDlg.h IpsecEntryDlg.cpp IpsecEntryDlg.h LifetimeEntryDlg.cpp LifetimeEntryDlg.h NetworkEntryDlg.cpp NetworkEntryDlg.h NodeEntryDlg.cpp NodeEntryDlg.h	nicht nennenswerte MFC Klassen	Für jeden Tabellen-Datensatz gibt es einen Dialog.
ConnectionDlg.cpp ConnectionDlg.h	nicht nennenswerte MFC Klassen	Connection-Übersichtsdialog
DlgGetPath.cpp DlgGetPath.h	nicht nennenswerte MFC Klassen	Pfadauswahl-Dialog (Source-Code aus dem Internet http://codeguru.earthweb.com/treeview/PathPicker.shtml)
PleaseWaitDlg.cpp PleaseWaitDlg.h	nicht nennenswerte MFC Klassen	Warte-Dialog
ColorListBox.cpp ColorListBox.h	nicht nennenswerte MFC Klassen	Farbige ListBox (Quellcode aus dem Internet http://codeguru.earthweb.com/listbox/colorlb.shtml)
DMGlobalDefines.h	-	Externe Deklarationen

DMTypes.cpp DMTypes.hpp	DMPathName	Spezielle Typen und eine Klasse
DMClasses.cpp DMClasses.hpp	DMNodeEntry DMNetworkEntry DMConnectionEntry DMIkeTransformEntry DMEspTransformEntry DMAhTransformEntry DMLifetimeEntry DMIpSecProposalEntry DMIkeProposalEntry DMNodeTableEntry DMNodeTableEntry DMNodeTableEntry DMIpsecProposalTableEntry DMIkeProposalTableEntry DMIkeTransformTableEntry DMEspTransformTableEntry DMAhTransformTableEntry DMLifetimeTableEntry	Von uns erweiterte Datentypen und Tabellen-Datensätze (ursprünglich Omnisec-spezifische Datentypen und Tabellen-Datensätze)
DMUtils.cpp DMUtils.hpp	-	Spezielle Funktionen (vor allem String Umwandlungen)
DMAlgorithms.cpp DMAlgorithms.hpp	-	Testalgorithmen für die Security Policies
DMMainAlgorithms.cpp DMMainAlgorithms.hpp	-	6 Funktionen welche die 6 Teststufen aus unserer Problemanalyse darstellen (zusammengesetzt aus den obigen Testalgorithmen)
SortedArray.h StdAfx.cpp StdAfx.h	nicht nennenswerte MFC Klassen	Von Visual C++ automatisch erzeugter Quellcode der in den Dialog-Klassen gebraucht wird

10.3.2.2 Klassendiagramm (Übersicht aller 64 Klassen)



10.3.2.3 Neue Klassen (gegenüber IpTables)

Klassen	Erklärung
DMNodeEntry DMNetworkEntry DMConnectionEntry DMIkeTransformEntry DMEspTransformEntry DMAhTransformEntry DMLifetimeEntry DMIpSecProposalEntry DMIkeProposalEntry	Unsere Datentypen. Die Basisklassen (NodeEntry, NetworkEntry, ...) wurden noch um verschiedene Klassenelemente erweitert, die die Resultate der Tests speichern.
DMNodeTableEntry DMNodeTableEntry DMNodeTableEntry DMIpsecProposalTableEntry DMIkeProposalTableEntry DMIkeTransformTableEntry DMEspTransformTableEntry DMAhTransformTableEntry DMLifetimeTableEntry	Unsere Tabellen-Datensätze (von TableEntry abgeleitet). Wir hatten zuerst diese Klassen von der entsprechenden Omnisec-Klasse abgeleitet (z.B. DMNodeTableEntry von NodeTableEntry abgeleitet), aber wir hatten Probleme mit den virtuell deklarierten Methoden (z.B. wurde beim Aufruf der Methode pack der Klasse DMNodeTableEntry nicht die überschriebene Methode writeToBuf von DMNodeEntry aufgerufen, sondern die writeToBufMethode der Basisklasse NodeEntry).
DMPathName	von Buffer_Base abgeleitete Klasse; Puffer für Pfadnamen eines Security Gateways
CAHTransformEntryDlg CAboutDlg CConnectionDlg CConnectionEntryDlg CDlgGetPath CESPTransformEntryDlg CExpertToolDlg CIkeEntryDlg CIkeTransformEntryDlg CIpsecEntryDlg CLifetimeEntryDlg CNetworkEntryDlg CNodeEntryDlg CPleaseWaitDlg CTablesDlg	Auf die von CDialog abgeleiteten Klassen werden wir nicht näher eingehen, da es sich hier um Klassen handelt, die von Visual C++ (genauer gesagt vom AppWizard) automatisch erzeugt werden. In der Visual C++ Hilfe findet man überdies eine ausführliche Beschreibung der Klasse CDialog.
CColorListBox	Klasse für Farbige ListBox (von CListBox abgeleitet; Quellcode aus dem Internet http://codeguru.earthweb.com/listbox/colorlb.shtml)
CExpertToolApp	Von Visual C++ automatisch generierte Klasse (abgeleitet von CWinApp, vgl. Hilfe in Visual C++)

10.3.2.4 Wichtige Erweiterungen der neuen Klassen

Klasse: DMNodeEntry

Variable	Bedeutung
ValidIP	Zeigt an, ob die IP-Adresse gültig ist
ValidNetworks	Zeigt an, ob alle Netzwerke dieses Security Gateways vom gleichen Typ sind (public / private)
ValidUniqueIP	Zeigt an, ob die IP-Address einmalig ist, oder ob sie in einem Netzwerk benützt wird
ValidSGNumbers[]	Speicher weitere Fehlerinformation
CompetingTablePath	Speichert den Pfad des Security Gateways, mit welchem dieser Eintrag ein Problem hat
CompetingTableEntry	Speichert den Index des konkurrierendem Eintrages
SameIPAddress	Zeigt an, ob die IP-Adresse die gleiche ist
SameOAD	Zeigt an, ob die OAD die gleiche ist

Klasse: DMNetworkEntry

Variable	Bedeutung
ValidIP	Zeigt an, ob die IP-Adresse gültig ist
ValidRange	Zeigt an, ob die Start-Adresse kleiner ist als die End-Adresse
ValidIPinRange	Zeigt an, ob alle IP-Adressen in dem Adress-Bereich gültig sind
ValidIndex0	Zeigt an, ob der Node-Index ungleich 0 ist
ValidIndexHigh	Zeigt an, ob der Node-Index nicht zu hoch ist
ValidNetwork[]	Speicher weitere Fehlerinformation
CompetingTablePath	Speichert den Pfad des Security Gateways, mit welchem dieser Eintrag ein Problem hat
CompetingTableEntry	Speichert den Index des konkurrierendem Eintrages
Overlapping	Zeigt an, ob die Netze überlappend sind
DoubleEntry	Zeigt an, ob es sich um doppelte Einträge handelt
SubNet	Zeigt an, ob es sich um ein Sub-Netz handelt, welches hinter einem anderen Security Gateway liegt

Klasse: DMConnectionEntry

Variabel	Bedeutung
ValidProt	Zeigt an, ob das Protokoll gültig ist
ValidIndexes	Zeigt an, ob alle Indizes ungleich 0 sind
ValidLocalRemoteNetwork	Zeigt an, ob das local und remote Netzwerk verschieden sind
ValidIndexesHigh[]	Zeigt an, ob die Indizes nicht zu hoch sind (local –net / remote-net / ike / ipsec)
NetworksBehind DifferentSG	Zeigt an, ob sich das local und remote Netzwerk hinter verschiedenen Security Gateways befinden
ValidConnection[]	Speicher weitere Fehlerinformation
CompetingTablePath	Speichert den Pfad des Security Gateways, mit welchem dieser Eintrag ein Problem hat
CompetingTableEntry	Speichert den Index des konkurrierendem Eintrages
DoubleEntry	Zeigt an, ob es sich um doppelte Einträge handelt
WrongOrder	Zeigt an, ob die Einträge in der falschen reihenfolge gespeichert sind
NoPartnerConnection	Zeigt an, ob die Verbindung im Ziel-Security Gateway keinen Eintrag hat
NoPartnerParameter	Zeigt an, ob die Verbindung im Ziel-Security Gateway keine gemeinsamen Parameter besitzt

Klasse: DMIkeTransformEntry

Variable	Bedeutung
ValidIke	Zeigt an, ob der Eintrag gültig ist

Klasse: DMEspTransformEntry

Variable	Bedeutung
ValidEsp	Zeigt an, ob der Eintrag gültig ist

Klasse: DMAhTransformEntry

Variable	Bedeutung
ValidAh	Zeigt an, ob der Eintrag gültig ist

Klasse: DMLifetimeEntry

Variable	Bedeutung
ValidLifeTime	Zeigt an, ob der Eintrag gültig ist

Klasse: DMIpSecProposalEntry

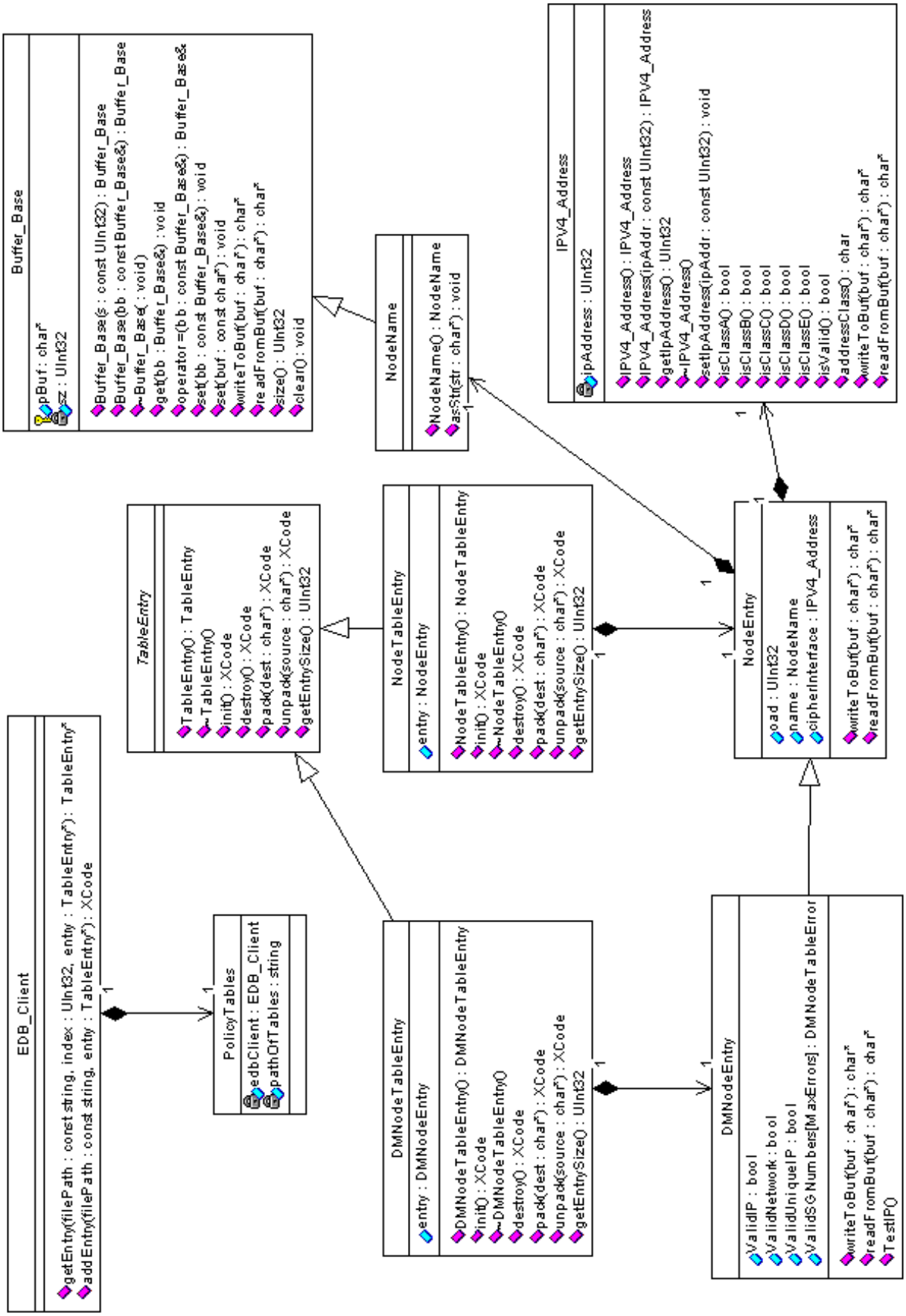
Variable	Bedeutung
ValidIndexes0	Zeigt an, ob die Indizes ungleich 0 sind
ValidIndexesHigh[]	Zeigt an, ob die Indizes nicht zu gross sind (ah / esp / lifetime)

Klasse: DMIkeProposalEntry

Variable	Bedeutung
ValidIndexes0	Zeigt an, ob die Indizes ungleich 0 sind
ValidIndexesHigh[]	Zeigt an, ob die Indizes nicht zu gross sind (lifetime / transform)

10.3.2.5 Erweitertes Klassendiagramm

(vgl. Kapitel 10.2.2.5)



Das erweiterte Klassendiagramm enthält nur zwei neue Klassen (vgl. Kapitel 10.2.2.5). Es sind dies die Klassen DMNodeEntry und DMNodeTableEntry. DMNodeEntry ist von NodeEntry abgeleitet. DMNodeTableEntry ist von TableEntry abgeleitet und enthält ein Klassenelement namens entry, das vom Klassentyp DMNodeEntry ist.

10.3.3 Bemerkungen zum Quellcode

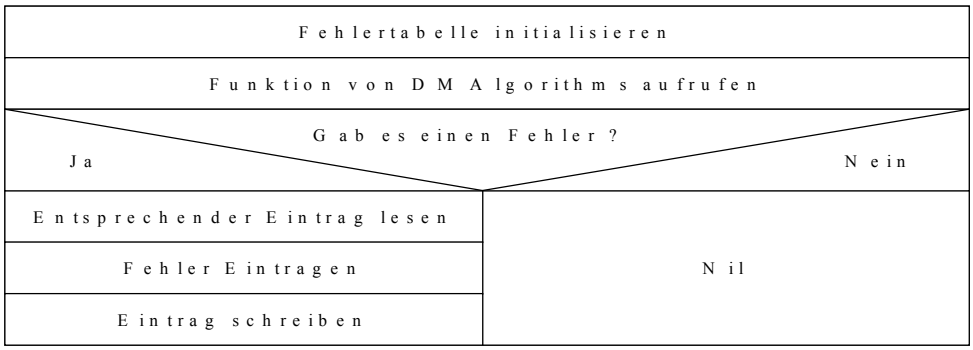
Wir werden nur einen kleinen Überblick über den Code geben. Für mehr Details befindet sich der kommentierte Source-Code im Anhang und auf der CD (vgl. Kapitel 15.3)

10.3.3.1 DMMainAlgorithms

In dieser Datei sind die Hauptfunktionen, welche die 6 verschiedenen Stufen testen. Die Funktionen wurden deshalb DMTestLvlOne bis DMTestLvlSix genannt.

DMTestLvlOne
In dieser Funktion erstellen wir für jede vorhandene Tabelle eine erweiterte Tabelle, in welcher wir die diversen Fehler speichern können. Zusätzlich werden auch noch alle Tests durchgeführt, welche zu der Stufe 1 gehören (Aufruf der verschiedenen Memberfunktionen der Klassen). Bei der Initialisierung werden die einzelnen Variablen als „kein Fehler“ gespeichert. Dies ermöglicht uns eine effizientere Abarbeitung der weiteren Test, da nur im Fehlerfall die Tabellen geändert werden müssen. Aus diesem Grund haben wir auch die Funktionen aus *DMAlgorithms.cpp* mit einem Rückgabewert versehen, welcher anzeigt, ob beim Testen ein Fehler erkannt wurde.

DMTestLvlTwo bis DMTestLvlSix
Diese Funktion sind alle ähnlich Strukturiert. Sie führen folgende Struktur für jeden Problemfall aus:



z.B. bei der Stufe 2 wird diese Struktur für jede Tabelle ausgeführt.

Die den beiden Funktionen DMTestLvlFive und DMTestLvlSix unterscheiden sich geringfügig, da diese nur einmal aufgerufen werden müssen, um die Tabellen von allen Security Gateways zu testen (die anderen Funktionen Testen immer nur die Tabellen von einem Security Gateway). Dadurch gibt es eine zusätzliche FOR-Schleife.

10.3.3.2 DMAlgorithms

In dieser Datei sind alle Funktionen enthalten, welche die verschiedenen Tabellen / Einträge / Werte testen.

Wir wollen folgenden Code-Ausschnitt, welcher in einigen der Funktionen benützt wird, genauer anschauen:

```
for (i = 1;i <= entries;i++){
  edbClient.getEntry(filename, i, &nodeTableEntry1);
  IPmain = nodeTableEfdfntry1.entry.cipherInterface.getIpAddress();
  OADmain = nodeTableEntry1.entry.oad;

    for (j = (i + 1);j <= entries;j++){
    edbClient.getEntry(filename, j, &nodeTableEntry2);
    IPtest = nodeTableEntry2.entry.cipherInterface.getIpAddress();
    OADtest = nodeTableEntry2.entry.oad;
```

Interessant ist die zweite „for"-Schleife. Durch den Startwert von „j = (i + 1)" werden nur die Einträge bearbeitet, welche sich tiefer in der Tabelle befinden. Dadurch kann sehr viel Rechenzeit gespart werden, da ja jedes Paar nur einmal getestet werden muss und nicht jeder Eintrag immer mit allen anderen. Um jedoch bei beiden Einträgen den Fehler zu speichern mussten wir danach folgenden Code einbauen:

```
if (temptest) {
if (errortable[i-1].ErrorCount<MaxErrors){
errortable[i-1].ErrorCount++;
errortable[i-1].ErrorNumber[errortable[i-1].ErrorCount1].CompetingTableEntry = j;
errortable[i-1].ErrorNumber[errortable[i-1].ErrorCount1].SameIPAddress = testIP;
errortable[i-1].ErrorNumber[errortable[i-1].ErrorCount-1].SameOAD = testOAD;
}//if
if (errortable[j-1].ErrorCount){
   errortable[j-1].ErrorCount++;
   errortable[j-1].ErrorNumber[errortable[j-1].ErrorCount-1].CompetingTableEntry = i;
   errortable[j-1].ErrorNumber[errortable[j-1].ErrorCount-1].SameIPAddress = testIP;
   errortable[j-1].ErrorNumber[errortable[j-1].ErrorCount-1].SameOAD = testOAD;
}//if
```

Wie man aus dem Code herauslesen kann, wird der Fehler zuerst in dem errortable-Eintrag des „Haupt" Eintrags und danach im errortable-Eintrag des zweiten Eintrags gespeichert. Dadurch ist auch im zweiten Eintrag vermerkt, mit welchem anderen Eintrag dieser ein Fehler hat. Das (-1) ist nötig, da in den Tabellen der erste Eintrag mit 1 und in der errortable der erste Eintrag mit 0 beginnt. Dies führte leider zu einigen Fehlern im Programm, welche wir danach mit viel Aufwand ausfindig machen mussten.

10.3.3.3 DMTypes

Wie der Name schon sagt, werden hier die verschiedenen von uns benötigten Typen definiert.

10.3.3.4 DMUtils

Hier sind alle Funktionen, welche keine Tests durchführen (z.B. DMGetEntryCount, welche die Einträge in einer Tabelle zählt).

Zwei Funktionen wollen wir speziell erwähnen:

XOR
Diese Funktion haben wir erstellt, weil wir ein logisches XOR benötigten, in C++ jedoch nur ein bit-weises xor vorhanden ist. Später erkannten wir, das wir auch das bit-weise xor verwenden könnten. Da aber das Programm funktionierte, wollten wir es nicht mehr ändern.

DMHexToASCII
Diese Funktion schrieben wir, weil wir keine Standart-Funktion kannten. Anstelle von „lange nach einer Funktion suchen“ entschieden wir uns, schnell eine eigene zu schreiben. Zu beachten gilt, das bei der „switch“ Anweisung die „break“ fehlen. Dies ist von uns so beabsichtigt, da wir die „switch“ Anweisung als „goto“ benützen.

10.3.3.5 DMClasses

Hier sind unsere erweiterten Klassen zu finden.

10.3.3.6 DMGlobalDefines

Definition der verschiedenen globalen Variablen.

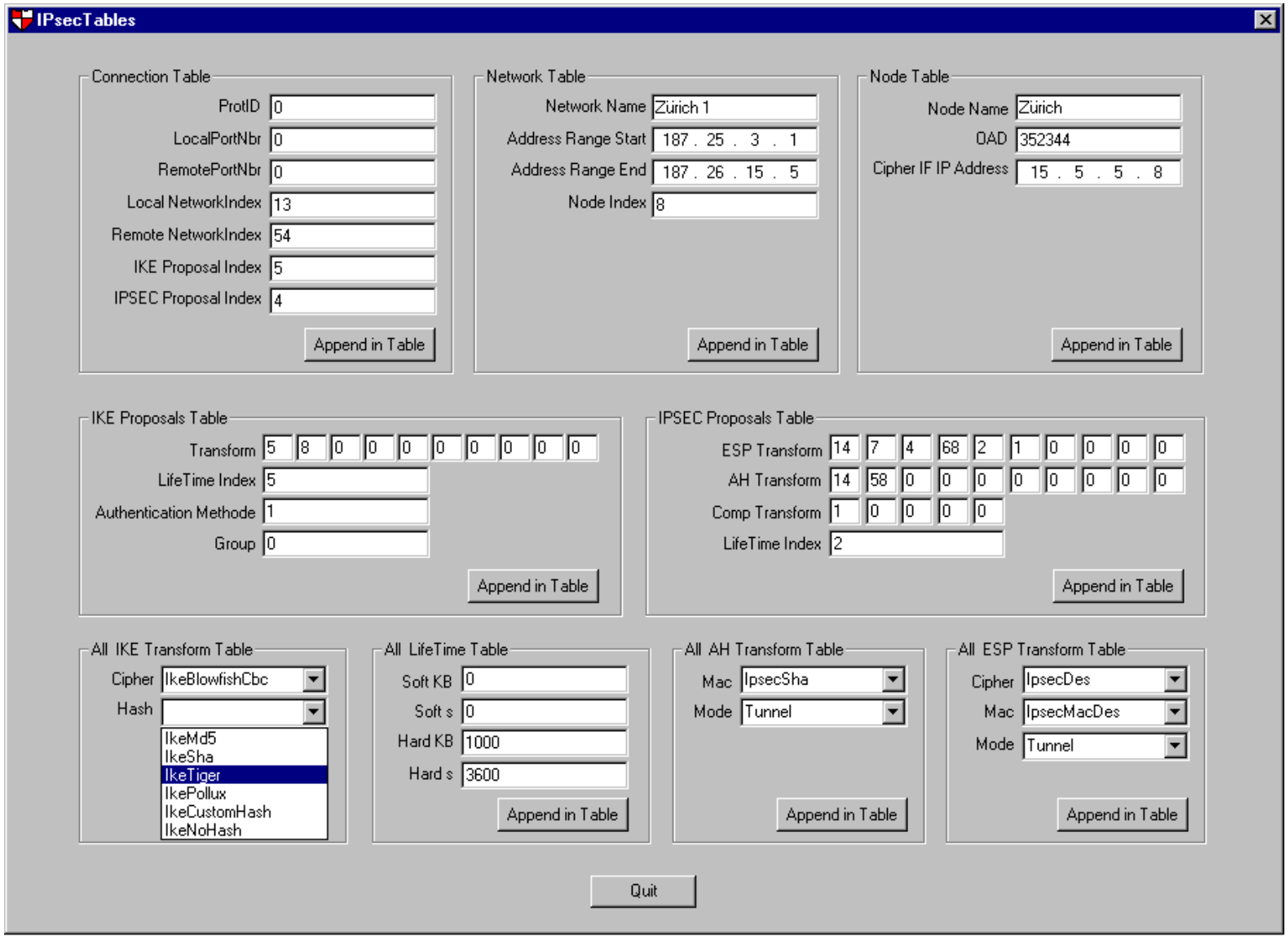
10.4 Programm "IpsecTables"

Diese Programm haben wir erstellt, damit wir für die Testzwecke die verschiedenen Tabellen schnell erstellen können. Dieses Tool wurde ausschliesslich für Selbstzwecke erstellt und war nicht ein Teil der Aufgabenstellung. Dadurch ist die Bedienerfreundlichkeit sehr gering. Z.B. werden die erstellten Tabellen im Verzeichnis „C:\TABLES“ gespeichert. Falls dieses Verzeichnis nicht vorhanden ist, werden die Tabellen nicht gespeichert. Es wird auch beim Einfügen eines Eintrags keine Meldung ausgegeben und die Felder werden nicht zurückgesetzt.

10.4.1 Benutzeranleitung

Die Bedienung ist sehr einfach. Für jede Tabelle gibt es einen eigenen Abschnitt. Zum Erstellen eines neuen Eintrags werden die dazugehörigen Felder ausgefüllt und der Knopf „Append in Table“ betätigt. Falls im Verzeichnis noch keine entsprechende Tabellen-Datei existiert, wird eine neue Datei erstellt, ansonsten wird der Eintrag am Ende der Datei angehängt. Für die Tabellen Ah-, Ike- und Esp-Transforms gibt es Combo-Boxen, mit welchen man die Verschiedenen Algorithmen auswählen kann. Bei den Tabellen IKE- und IPSEC- proposal müssen alle Felder ausgefüllt sein (Ah-Transform, Esp-Transform, u.s.w. Bei den nicht benötigten Feldern eine 0 einfügen (siehe folgendes Bild)).

Nachfolgend ist ein Abbild der Benutzeroberfläche.


IPsecTables
Connection Table
ProtID 0
LocalPortNbr 0
RemotePortNbr 0
Local NetworkIndex 13
Remote NetworkIndex 54
IKE Proposal Index 5
IPSEC Proposal Index 4
Append in Table
Network Table
Network Name Zürich 1
Address Range Start 187 . 25 . 3 . 1
Address Range End 187 . 26 . 15 . 5
Node Index 8
Append in Table
Node Table
Node Name Zürich
OAD 352344
Cipher IF IP Address 15 . 5 . 5 . 8
Append in Table
IKE Proposals Table
Transform 5 8 0 0 0 0 0 0 0 0
LifeTime Index 5
Authentication Methode 1
Group 0
Append in Table
IPSEC Proposals Table
ESP Transform 14 7 4 68 2 1 0 0 0 0
AH Transform 14 58 0 0 0 0 0 0 0 0
Comp Transform 1 0 0 0 0
LifeTime Index 2
Append in Table
All IKE Transform Table
Cipher IkeBlowfishCbc
Hash
IkeMd5
IkeSha
IkeTiger
IkePollux
IkeCustomHash
IkeNoHash
All LifeTime Table
Soft KB 0
Soft s 0
Hard KB 1000
Hard s 3600
Append in Table
All AH Transform Table
Mac IpsecSha
Mode Tunnel
Append in Table
All ESP Transform Table
Cipher IpsecDes
Mac IpsecMacDes
Mode Tunnel
Append in Table
Quit

10.4.2 Softwaredesign

Auf das Programm IPsecTables werden wir an dieser Stelle nicht mehr genauer eingehen. Dies aus 2 Gründen:

- Dieses Programm gehört nicht zur Aufgabenstellung dieser Diplomarbeit und wurde ausschliesslich für Selbstzwecke erstellt (um Test-Tabellen zu erzeugen).
- Vom Design her ist es praktisch identisch zum Programm IpTables, das wir weiter vorn schon ausführlich erklärt haben. Der einzige Unterschied ist, dass unser Programm eine GUI hat, die aber sehr einfach programmiert ist.

11 Tests

11.1 Allgemeines

Um mit unserem Tool alle Algorithmen zu testen, mussten wir 3 Datensätze erstellen. Der erste Datensatz ist Fehlerfrei. Beim 2. Datensatz haben wir Fehler eingebaut, welche den Problemen der ersten 5 Stufen entsprechen (siehe Kapitel 7 Problemanalyse). Da das Tool für die Stufe 6 fehlerfreie Tabellen benötigt (Stufe 1-5), mussten wir den 3 Datensatz erstellen, welcher nur Probleme der Stufe 6 enthält. Da die erstellten Testtabellen sehr viel Rechenzeit benötigen um getestet zu werden, habe wir kleinere Tabellen erstellt, welche wir benützten, um die letzten Fehler im Tool zu lokalisieren und zu beheben. Diese Tabellen werden wir nicht aufführen, sie sind jedoch auf der CD (siehe Kapitel 15.3) gespeichert

11.2 Datensatz 1: Fehlerfrei

(TablesNoErrors.zip)
Um auch gleich die Speicherbedürfnisse und die Rechenzeit zu testen, haben wir uns entschieden, relativ grosse Tabellen zu erstellen. Wir haben ein Netzwerk mit 20 Security Gateways, 56 Netzwerke und 100 verschiedene Verbindungen. Um die Tabellen zu erstellen, haben wir uns ein kleines Tool geschrieben, mit welchem man sehr schnell die Tabellen erstellen kann (IpsecTables.exe). Um das erstellen der Tabellen zu verkürzen, haben wir uns entschieden, in jedem Security Gateway die gleichen Tabellen zu benützen (ausser der Tabelle mit den Verbindungen). Dadurch wird zwar mehr Speicherplatz und mehr Rechenzeit benötigt, dafür ging das erstellen der Tabellen schneller und mit weniger Fehler. Die Tabellen sind im Anhang 15.4 aufgeführt. In der „Connection-Table“ sind alle 100 Verbindungen zusammengefasst und nicht für jeden Security Gateway eine eigene Tabelle aufgeführt (wie es bei dem Datensatz gelöst ist).

11.3 Datensatz 2: Fehler Stufe 1-5

(Tableslvl1to5.zip)
Der Datensatz ist eine Erweiterung des ersten Datensatzes. Um die Änderungen durchzuführen, haben wir das Programm „Microsoft Visual C++ 6.0“ verwendet. Das Programm erlaubte uns, einzelne Bytes in den Dateien zu ändern. Falls wir einen ganzen Eintrag einfügen mussten, haben wir unser Tabellen-Tool (IpsecTables.exe) verwendet.

Die Tabellen wurden wie folgt verändert / erweitert:

Im Security Gateway 00-00-00-7B wurden Änderungen an der Node Table vorgenommen (**FETT**):

Tabellen Index	SG Name	OAD	IP-Adresse
1	Zürich	**FF-01-E2-40**	**100.0.0.2**
2	Bern	**00-00-15-38**	**100.0.1.2**
3	Basel	**FF-00-01-B0**	**255.255.255.255**
4	Baden	**FF-00-10-75**	**0.0.0.0**
5	Chur	**FF-00-15-38**	**127.0.0.1**
6	Zug	**FF-00-02-22**	**90.2.2.10**
7	Schlieren	**FF-00-09-27**	**192.168.0.14**

Zuerst haben wir in jedem Eintrag nur die OAD oder die IP-Adresse geändert. Wir haben uns nach dem ersten Testlauf jedoch dafür entschieden, beide Werte zu ändern, da es zu viele Fehler gab (Bsp. Eintrag 4 (Baden): durch das Ändern der IP-Adresse von 100.0.0.4 zu 0.0.0.0 entsteht der gewünschte Fehler (ungültige IP-Adresse), jedoch hat dieser Eintrag nun bei allen andern Tabellen ein Problem mit dem Eintrag 4, da die OAD gleich ist, die IP-Adresse aber nicht.
Folgende Probleme können mit dieser Tabelle getestet werden :

- doppelte IP
- doppelte OAD
- ungültige IP-Adresse
- IP-Adresse in einem Netzwerk enthalten
- IP-Adresse ist Privat

Das Tool hat alle Fehler Erkannt.

Im Security Gateway 00-00-01-B0 wurden Änderungen an der Network Table vorgenommen (**FETT**):

Tabellen Index	Name	IP- Start	IP- Ende	Node Index
1	Zürich 1	**1.0.0.254**	**1.0.0.1**	1
2	Zürich 2	**127.0.0.1**	**127.0.0.10**	1
3	Zürich 3	**126.0.0.1**	**128.0.0.1**	1
4	Bern 1	80.0.0.1	80.0.0.254	**0**
5	Basel 1	192.0.12.1	192.0.12.20	**80**
57	**Moskau 3**	**10.0.0.1**	**10.0.0.10**	**20**
58	**Moskau 4**	**90.6.0.1**	**90.6.255.254**	**20**

Folgende Probleme können mit dieser Tabelle getestet werden:
- IP-End-Adresse kleiner als IP-Start-Adresse
- IP-Adresse ungültig
- Ungültige IP-Adresse(n) in der „Address-Range“
- Node-Index = 0
- Node-Index zu gross
- Privat und Public Netzwerke hinter dem gleichen Security Gateway

Das Tool hat alle Fehler Erkannt.

Im Security Gateway 00-00-02-22 wurden Änderungen an der Lifetime Table vorgenommen (**FETT**):

Tabellen Index	Soft kByte	Soft Sekunden	Hard kByte	Hard Sekunden
4	**0**	**0**	**0**	**0**
5	**2000**	**3000**	**1000**	**2000**

Folgende Probleme können mit dieser Tabelle getestet werden:
- Hard kByte = 0
- Hard Sekunden = 0
- Soft kByte > Hard kByte
- Soft Sekunden > Hard Sekunden

Das Tool hat alle Fehler Erkannt.

Im Security Gateway 00-00-02-34 wurden Änderungen an den Transform Tables vorgenommen (**FETT**):

AH-Transform

Tabellen Index	Mac	Mode
1	IpsecMd5	**??? (Value 3)**
2	**??? (Value 15)**	Transport

ESP-Transform

Tabellen Index	Cipher	Mac	Mode
1	IpsecDesIv64	IpsecMd5	**??? (Value 3)**
2	IpsecDesIv64	**??? (Value 15)**	Transport
3	**??? (Value 15)**	IpsecMacDes	Tunnel

IKE-Transform

Tabellen Index	Cipher	Hash
1	IkeDesCbc	**??? (Value 7)**
2	**??? (Value 15)**	IkeMd5

Folgende Probleme können mit dieser Tabelle getestet werden:
- ungültiger Index AH-Mode
- ungültiger Index AH-Mac
- ungültiger Index ESP-Mode
- ungültiger Index ESP-Mac
- ungültiger Index ESP-Cipher
- ungültiger Index IKE-Hash
- ungültiger Index IKE-Cipher

Das Tool hat alle Fehler Erkannt.

Im Security Gateway 00-00-09-26 wurden Änderungen an den Proposal Tables vorgenommen (**FETT**):

IKE Proposal

Tabellen Index	Transform	Lifetime	Authentication Methode*	Group*
10	**0**	**7**	**1**	**2**
11	**34,23,1,32,4,2**	**2**	**1**	**0**

IPsec Proposal

Tabellen Index	Esp Transform	Ah Transform	Comp Transform*	Lifetime
13	**34,32,1,3**	**56**	**1**	**10**
14	**0**	**0**	**1**	**1**

*werden in dieser Version nicht getestet

Folgende Probleme können mit dieser Tabelle getestet werden:
- Indizes zu gross
- Indizes = 0

Das Tool hat alle Fehler Erkannt.

Im Security Gateway 00-00-10-75 wurden Änderungen an der Connection Table vorgenommen (**FETT**):

Local port	Remote port	Local net	Remote net	Ike prop	Ipsec prop
0	**0**	**10**	**10**	**1**	**1**
0	**0**	**12**	**8**	**2**	**2**
0	**0**	**12**	**9**	**2**	**2**
0	**0**	**49**	**0**	**20**	**35**
0	**0**	**12**	**8**	**2**	**2**
20	**0**	**10**	**10**	**1**	**1**
0	**0**	**1**	**2**	**3**	**4**

Folgende Probleme können mit dieser Tabelle getestet werden:
- Ungültiges Protokoll
- Doppelter Eintrag
- Local und Remote Netzwerk das gleiche
- Local und Remote Netzwerk hinter gleichem Security Gateway
- Zu grosse Indizes
- Indizes = 0
- Falsche Reihenfolge (Ports)
- Falsche Reihenfolge (Sub-Net)

Das Tool hat alle Fehler Erkannt.

11.4 Datensatz 3: Fehler Stufe 6

(Tableslvl6.zip)
Für diesen Datensatz haben wir wieder den Datensatz 1 als Grundlage verwendet und einige kleine Änderungen vorgenommen. Wir haben bei einer Verbindung das IKE-Proposal so abgeändert, dass es mit dem Eintrag im „Remote Security Gateway“ keine Übereinstimmung mehr gibt. Wir haben auch eine neue Verbindung Erstellt, welche nur in einem Security Gateway gespeichert ist.

Im Security Gateway 00-00-00-7B wurden Änderungen an der Connection Table vorgenommen (**FETT**):

Tabellen Index*	Prot ID	Local port	Remote port	Local net	Remote net	Ike prop	Ipsec prop
7	0	0	0	4	34	**6**	7
101	**0**	**0**	**0**	**4**	**1**	**1**	**1**

*Index bezieht sich auf die Tabelle mit allen Verbindungen und nicht auf die Tabelle des Security Gateway 00-00-00-7B

Folgende Probleme können mit dieser Tabelle getestet werden:
- Keine Gegenverbindung
- Gegenverbindung hat keine gemeinsamen Parameter (IKE-Transform, Lifetime,...)

Das Tool hat alle Fehler Erkannt.

12 Probleme

12.1 Virtuelle Element Funktionen

Wir hatten zuerst folgende Klassen von der entsprechenden Omnisec-Klasse abgeleitet (z.B. DMNodeTableEntry von NodeTableEntry abgeleitet):

- DMNodeTableEntry
- DMNodeTableEntry
- DMNodeTableEntry
- DMIpsecProposalTableEntry
- DMIkeProposalTableEntry
- DMIkeTransformTableEntry
- DMEspTransformTableEntry
- DMAhTransformTableEntry
- DMLifetimeTableEntry

Wir hatten jedoch Probleme mit den virtuell deklarierten Methoden. Z.B. wurde beim Aufruf der Methode pack der Klasse DMNodeTableEntry nicht die überschriebene Methode writeToBuf von DMNodeEntry aufgerufen, sondern die writeToBufMethode der Basisklasse NodeEntry. Daher leiteten wir unsere Klassen direkt von TableEntry ab.

12.2 Pointer

Wir hatten mit einigen Pointern das Problem, dass sie nicht absturzfrei gelöscht werden konnten. Daher lassen wir sie in unserem Programm "weiterleben". Dies führt dazu das der Speicherbedarf unseres Programms dauernd ein wenig am wachsen ist, so lange es noch läuft. Beim Beenden des Programms wird der Speicher natürlich wieder freigegeben. Da unser Programm keine Serveranwendung ist, sollte es keine Speicherprobleme bereiten. Wir wissen, dass dies kein sauberer Programmierstil ist, konnten aber aus Zeitgründen nicht mehr die Ursache dieses Problems eruieren.

12.3 Bekannter Bug

In der Tabellenansicht (Tablesview) in der Tabelle NodeTableEntries werden Fehler zum Teil mehrfach aufgeführt. Leider blieb uns keine Zeit mehr, um diesen Fehler zu beheben.

13 Verbesserungen / Erweiterungen

Wir sind mit der von uns Erstellten Software zufrieden, wie jedoch bei jedem Softwareprodukt gibt es auch bei unserem noch nützliche Verbesserungen und Erweiterungen.

13.1 Verbesserungen

- Implementierung der Fehlenden Algorithmen
- Bestehende Algorithmen optimieren
- Entdeckte Fehler genauer Umschreiben
- Nach dem Beenden des Programms die erweiterten Tabellen wieder löschen
- Erwähnte Bugs und Pointerprobleme beheben

13.2 Erweiterungen

- Editierfunktionen um die fehlerhafte Einträge gleich korrigieren zu können
- Download und Upload der Tabellen / korrigierten Tabellen in Tool integrieren
- Ausdruck eines Log-Files

14 Schlusswort

Unsere Diplomarbeit hat eigentlich mehr mit Software zu tun gehabt als mit Kommunikation. Dies war jedoch mit ein Grund wieso wir uns für diese Arbeit entschieden. Ein anderer Grund war das interessante Thema Internetsicherheit.

Diese Diplomarbeit war eine grosse Herausforderung für uns. Wir sammelten wertvolle Erfahrungen im Bereich Softwaredesign und Fehlereingrenzung. Die Teamarbeit gestaltete sich höchst erfreulich, weil wir uns gut ergänzt haben. Wir sind mit dem Ergebnis der geleisteten Arbeit sehr zufrieden.

Wir bedanken uns bei David Vonarburg und Peter Fernandez der Firma Omnisec für die gute Kooperation. Dank geht auch an Herrn Dr. Andreas Steffen für die gute Betreuung.

15 Anhang

15.1 Glossar

AH	Authentication Header
CHAP	Challenge Handshake Authentication Protocol
COMP	Compression
DM	D'Aquino und Meier
EDB	Embedded Database
ESP	Encapsulating Security Payload
GUI	Graphical User Interface
IKE	Internet Key Exchange
IO	Input/Output
IP	Internet Protocol
IPSEC	IP-Security
IPX	Internetwork Packet eXchange
ISP	Internet Service Provider
MFC	Microsoft Foundation Class Library
OAD	Eineindeutige Identifizierungsnummer
OAG	Omnisec AG
PAP	Password Authentication Protocol
S-HTTP	Secure Hyper Text Transfer Protocol
S-MIME	Secure Multipurpose Internet Mail Extension
SOCKS	Socket Secure Server
SPD	Security Policy Database
SSL	Secure Socket Layer
TCP	Transport Control Protocol
UDP	User Datagram Protocol
VPN	Virtual Private Network

15.2 Quellenverzeichnis

15.2.1 Literatur

Dave Kosiur, "Building and Managing Virtual Private Network", John Wiley & Sons, Toronto, 1998
ISBN 0-471-29526-4

Andrew S. Tanenbaum, "Computernetzwerke", Prentice Hall, München, 1997
ISBN 3-8272-9536-X

15.2.2 Diplomarbeiten

Oliver Gärtner und Berkant Uenal, "Virtual Private Network mit sicherem Tunnel durchs Internet", Nummer Sna 99/1, Winterthur, 1999

15.2.3 Internet

http://codeguru.earthweb.com/listbox/colorlb.shtml
Quellcode und Erklärungen zur Klasse CColorListBox

http://codeguru.earthweb.com/treeview/PathPicker.shtml
Quellcode und Erklärungen zur Klasse CDlgGetPath

http://cespc1.kumoh.ac.kr/~sylot/etc/ascii.html
ASCII-Code-Tabelle

http://search.microsoft.com/us/dev/default.asp
MSDN Online Search (mehr Hilfe zu Visual C++)

http://www.rational.com/index.jsp
Homepage von Rational Rose (dient unter anderem zur Erstellung von Klassendiagrammen)

http://www.codevizor.com/
Homepage von Programm CodeVizor (dient zur Erstellung von Klassendiagrammen)

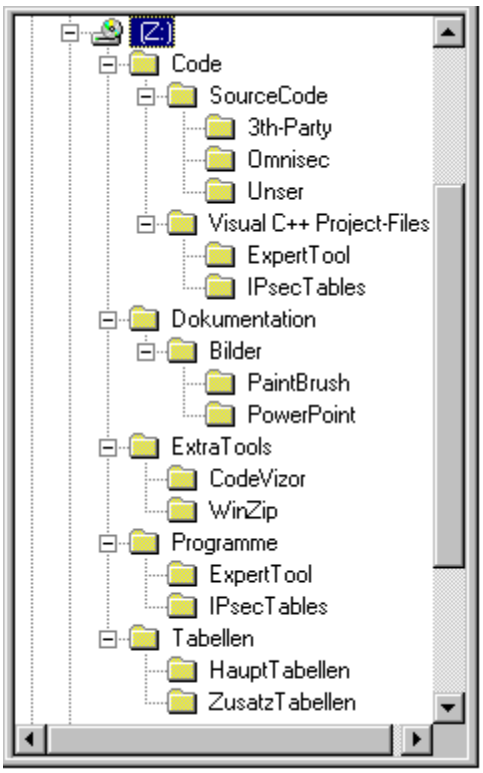
http://www.winzip.com/
Homepage von Winzip

http://www.omnisec.ch/
Homepage unseres Industriepartners Omnisec AG

15.3 Inhalt der CD

Verzeichnis	Inhalt
Code	Hier befinden sich die verschiedenen Code-Files
SourceCode	Alle Source-Code Files (cpp+hpp+h)
3th-Party	Module, welche wir vom Internet bezogen haben
Omnisec	Die Dateien, welche uns von Omnisec zur Verfügung gestellt wurden
Unser	Die Dateien, welche wir erstellt haben
Visual C++ Project-Files	Die Kompletten Visual C++ Projekte
ExpertTool	Das Projekt „ExpertTool“
IPsecTables	Das Projekt „IpsecTables“
Dokumentation	Die Dokumentation
Bilder	Die Bilder, welche in unserer Dokumentation verwendet werden
PaintBrush	Die Bilder im bmp Format
PowerPoint	Die Bilder im ppt Format
ExtraTools	Extra Programme, welche wir benötigten
CodeVizor	Tool zum erstellen von Klassen-Diagrammen
WinZip	Zum extrahieren der Test-Tabellen
Programme	Die erstellten Programme (exe)
ExpertTool	ExpertTool.exe
IPsecTables	IpsecTables.exe
Tabellen	Die Tabellen, die wir für die Testzwecke erstellt haben (zip-Format)
HauptTabellen	Die Tabellen für den Funktionalitäts-Test
ZusatzTabellen	Die Tabellen, welche wir zum Debuggen benützten

Nachfolgend ein Bild, welches die Struktur im Windows NT-Explorer darstellt:



15.4 Test-Tabellen

15.4.1 IKE-Transform

Tabellen Index	Cipher	Hash
1	IkeDesCbc	IkeMd5
2	IkeIdeaCbc	IkeMd5
3	IkeBlowfishCbc	IkeMd5
4	IkeRc5R16B64Cbc	IkeMd5
5	IkeDes3Cbc	IkeMd5
6	IkeCastCbc	IkeMd5
7	IkeDesCbc	IkeSha
8	IkeIdeaCbc	IkeSha
9	IkeBlowfishCbc	IkeSha
10	IkeRc5R16B64Cbc	IkeSha
11	IkeDes3Cbc	IkeSha
12	IkeCastCbc	IkeSha
13	IkeDesCbc	IkeTiger
14	IkeIdeaCbc	IkeTiger
15	IkeBlowfishCbc	IkeTiger
16	IkeRc5R16B64Cbc	IkeTiger
17	IkeDes3Cbc	IkeTiger
18	IkeCastCbc	IkeTiger

15.4.2 ESP-Transform

Tabellen Index	Cipher	Mac	Mode
1	IpsecDesIv64	IpsecMd5	Tunnel
2	IpsecDesIv64	IpsecSha	Transport
3	IpsecDes	IpsecMacDes	Tunnel
4	IpsecDes	IpsecMd5	Transport
5	IpsecDes3	IpsecSha	Tunnel
6	IpsecDes3	IpsecMacDes	Transport
7	IpsecRc5	IpsecMd5	Tunnel
8	IpsecRc5	IpsecSha	Transport
9	IpsecIdea	IpsecMacDes	Tunnel
10	IpsecIdea	IpsecMd5	Transport
11	IpsecCast	IpsecSha	Tunnel
12	IpsecCast	IpsecMacDes	Transport
13	IpsecBlowfish	IpsecMd5	Tunnel
14	IpsecBlowfish	IpsecSha	Transport
15	IpsecIdea3	IpsecMacDes	Tunnel
16	IpsecIdea3	IpsecMd5	Transport
17	IpsecDesIv32	IpsecSha	Tunnel
18	IpsecDesIv32	IpsecMacDes	Transport
19	IpsecRc4	IpsecMd5	Tunnel
20	IpsecRc4	IpsecSha	Transport

15.4.3 AH-Transform

Tabellen Index	Mac	Mode
1	IpsecMd5	Tunnel
2	IpsecMd5	Transport
3	IpsecSha	Tunnel
4	IpsecSha	Transport
5	IpsecMacDes	Tunnel
6	IpsecMacDes	Transport

15.4.4 Life Time

Tabellen Index	Soft kByte	Soft Sekunden	Hard kByte	Hard Sekunden
1	0	0	1000	3600
2	500	3200	1000	3600
3	1000	3600	2000	7200

15.4.5 IKE- Proposal

Tabellen Index	Transforms	Lifetime	Authentication Methode	Group
1	1,2,3,4,5	1	1	0
2	2,4,6,8	2	1	0
3	1,2,3,4,5,6,12,13,15	3	1	0
4	18,4,2,6,3,16,12	1	1	0
5	3,5,2	2	1	0
6	11	3	1	0
7	14,15	1	1	0
8	9,7,13	2	1	0
9	7	3	1	0

15.4.6 IPsec- Proposal

Tabellen Index	Esp Transform	Ah Transform	Comp Transform	Life Time
1	1,2,3,4,5	1,2,3	1	1
2	20,1,3,17,5,9,10	2,4,6	1	2
3	2,14,1,9,10,3,7,8	3,6	1	3
4	10,12,14,16,18,20	4	1	1
5	1,3,5,7,9,11,13,15	5,4,2,1	1	2
6	3,2,1	6,1	1	3
7	1,4,5,6	1,5	1	1
8	11,13,17,14,5,1,8,3	2,4,3,1	1	2
9	4,12,16,8,3,7	3,6,5,4	1	3
10	1,2,3,4,5,6,7,8,9,10	4,1,2	1	1
11	3,2,1,5,6,7	5,3,1	1	2
12	6,5,4,3,2,1	6,4	1	3

15.4.7 SGs

Tabellen Index	SG Name	OAD	IP-Addresse
1	Zürich	00-01-E2-40	100.0.0.1
2	Bern	00-00-00-7B	100.0.0.2
3	Basel	00-00-01-B0	100.0.0.3
4	Baden	00-00-10-75	100.0.0.4
5	Chur	00-00-15-38	100.0.0.5
6	Zug	00-00-02-22	100.0.0.6
7	Schlieren	00-00-09-27	100.0.0.7
8	Weinfelden	00-00-02-34	100.0.0.8
9	Winterthur	00-00-1D-E5	100.0.0.9
10	Altstetten	00-00-64-99	100.0.0.10
11	Buchs	00-00-5C-0B	100.0.0.11
12	Sargans	00-00-09-26	100.0.0.12
13	Wien	00-00-15-A8	100.0.0.13
14	Berlin	00-00-1E-C3	100.0.0.14
15	London	00-00-11-D8	100.0.0.15
16	Paris	00-00-5B-4F	100.0.0.16
17	New York	20-64-7D-20	100.0.0.17
18	Washington	1A-07-1B-B9	100.0.0.18
19	Tokio	0F-39-CA-96	100.0.0.19
20	Moskau	01-67-4A-44	100.0.0.20

15.4.8 Netzwerke

Tabellen Index	Name	IP- Start	IP- Ende	Node Index
1	Zürich 1	1.0.0.1	1.0.0.254	1
2	Zürich 2	11.0.0.1	11.0.0.254	1
3	Zürich 3	111.0.0.1	111.0.0.254	1
4	Bern 1	80.0.0.1	80.0.0.254	2
5	Basel 1	192.0.12.1	192.0.12.20	3
6	Basel 2	192.0.12.21	192.0.12.30	3
7	Baden 1	223.0.0.1	223.0.0.20	4
8	Chur 1	130.0.5.1	130.0.5.20	5
9	Chur 2	130.0.5.1	130.0.5.1	5
10	Chur 3	130.0.5.21	130.0.5.30	5
11	Zug 1	223.0.1.1	223.0.1.20	6
12	Zug 2	223.0.1.21	223.0.1.40	6
13	Schlieren 1	10.0.0.1	10.0.255.254	7
14	Schlieren 2	10.1.0.1	10.1.255.254	7
15	Schlieren 3	10.2.0.1	10.2.255.254	7
16	Schlieren 4	10.3.0.1	10.3.255.254	7
17	Schlieren 5	10.4.0.1	10.4.255.254	7
18	Schlieren 6	10.5.0.1	10.5.255.254	7
19	Weinfelden 1	200.200.200.1	200.200.200.20	8
20	Weinfelden 2	200.200.200.21	200.200.200.40	8
21	Winterthur 1	191.0.0.1	191.0.0.254	9
22	Winterthur 2	191.0.1.1	191.0.1.254	9
23	Winterthur 3	191.0.2.1	191.0.2.254	9
24	Winterthur 4	191.0.3.1	191.0.3.254	9
25	Winterthur 5	191.0.4.1	191.0.4.254	9
26	Altstetten 1	192.0.0.1	192.0.0.40	10
27	Altstetten 2	192.0.0.2	192.0.0.2	10
28	Buchs 1	90.0.0.1	90.0.0.124	11
29	Buchs 2	90.0.0.125	90.0.0.254	11
30	Sargans 1	90.0.1.1	90.0.1.254	12
31	Wien 1	90.0.2.1	90.0.2.80	13
32	Wien 2	90.0.2.81	90.0.2.160	13
33	Wien 3	90.0.2.161	90.0.2.200	13
34	Wien 4	90.0.2.201	90.0.2.254	13
35	Berlin 1	90.0.3.1	90.0.3.254	14
36	Berlin 2	90.0.4.1	90.0.4.254	14
37	Berlin 3	90.0.5.1	90.0.5.254	14
38	Berlin 4	90.0.6.1	90.0.6.254	14
39	London 1	90.0.7.1	90.0.7.254	15
40	London 2	90.0.8.1	90.0.8.254	15
41	London 3	90.0.9.1	90.0.9.254	15
42	London 4	90.0.10.1	90.0.10.254	15
43	Paris 1	90.0.11.1	90.0.254.254	16
44	Paris 2	90.1.1.1	90.1.254.254	16
45	New York 1	90.2.0.1	90.2.0.254	17
46	New York 2	90.2.1.1	90.2.1.254	17
47	New York 3	90.2.2.1	90.2.2.254	17
48	New York 4	90.2.3.1	90.2.3.254	17

49	Washington 1	90.2.4.1	90.2.4.254	18
50	Washington 2	90.2.5.1	90.2.5.254	18
51	Tokio 1	90.3.0.1	90.3.255.254	19
52	Tokio 2	90.4.0.1	90.4.255.254	19
53	Tokio 3	90.5.0.1	90.5.255.254	19
54	Tokio 4	90.6.0.1	90.6.255.254	19
55	Moskau 1	90.7.0.1	90.7.0.254	20
56	Moskau 2	90.7.1.1	90.7.1.254	20

15.4.9 Connection

Tabellen Index	Prot	Port a	Port b	Netz a	Netz b	Ike	IPsec
1	0	0	0	1	28	1	1
2	0	0	0	1	29	2	2
3	0	0	0	2	30	3	3
4	0	0	0	2	31	4	4
5	0	0	0	3	32	5	5
6	0	0	0	3	33	6	6
7	0	0	0	4	34	7	7
8	0	0	0	4	35	8	8
9	0	0	0	5	36	9	9
10	0	0	0	5	37	1	10
11	0	0	0	6	38	2	11
12	0	0	0	6	39	3	12
13	0	0	0	7	40	4	1
14	0	0	0	7	41	5	2
15	0	0	0	8	42	6	3
16	0	0	0	8	43	7	4
17	0	0	0	9	44	8	5
18	0	0	0	9	45	9	6
19	0	0	0	10	46	1	7
20	0	0	0	10	47	2	8
21	0	1	0	11	48	3	9
22	0	2	0	11	49	4	10
23	0	3	0	12	50	5	11
24	0	4	0	12	51	6	12
25	0	5	0	13	52	7	1
26	0	6	0	13	53	8	2
27	0	7	0	14	54	9	3
28	0	8	0	14	55	1	4
29	0	9	0	15	56	2	5
30	0	10	0	15	28	3	6
31	0	11	0	16	29	4	7
32	0	12	0	16	30	5	8
33	0	13	0	17	31	6	9
34	0	14	0	17	32	7	10
35	0	15	0	18	33	8	11
36	0	16	0	18	34	9	12
37	0	17	0	19	35	1	1
38	0	18	0	19	36	2	2
39	0	19	0	20	37	3	3
40	0	20	0	20	38	4	4
41	0	0	1	21	39	5	5
42	0	0	2	21	40	6	6
43	0	0	3	22	41	7	7
44	0	0	4	22	42	8	8
45	0	0	5	23	43	9	9
46	0	0	6	23	44	1	10
47	0	0	7	24	45	2	11
48	0	0	8	24	46	3	12
49	0	0	9	25	47	4	1

50	0	0	10	25	48	5	2
51	0	0	11	26	49	6	3
52	0	0	12	26	50	7	4
53	0	0	13	27	51	8	5
54	0	0	14	27	52	9	6
55	0	0	15	28	2	1	7
56	0	0	16	28	9	2	8
57	0	0	17	29	13	3	9
58	0	0	18	29	24	4	10
59	0	0	19	30	8	5	11
60	0	0	20	30	10	6	12
61	0	5	5	31	14	7	1
62	0	5	5	31	15	8	2
63	0	5	5	32	23	9	3
64	0	5	5	32	24	1	4
65	0	5	5	33	16	2	5
66	0	6	6	33	17	3	6
67	0	6	6	34	6	4	7
68	0	6	6	34	7	5	8
69	0	6	6	35	9	6	9
70	0	6	6	35	8	7	10
71	0	6	6	36	10	8	11
72	0	11	0	36	11	9	12
73	0	11	0	37	12	1	1
74	0	11	0	37	13	2	2
75	0	11	0	38	14	3	3
76	0	11	0	38	15	4	4
77	0	11	11	39	16	5	5
78	0	11	11	39	17	6	6
79	0	11	11	40	18	7	7
80	0	11	11	40	19	8	8
81	0	1	1	41	20	9	9
82	0	1	1	41	21	1	10
83	0	1	1	42	10	2	11
84	0	1	1	42	11	3	12
85	0	1	1	43	12	4	1
86	0	1	1	43	13	5	2
87	0	1	1	44	14	6	3
88	0	11	0	44	15	7	4
89	0	1	1	45	16	8	5
90	0	1	1	46	17	9	6
91	0	1	1	47	18	1	7
92	0	1	1	48	19	2	8
93	0	1	1	49	20	3	9
94	0	0	0	50	21	4	10
95	0	1	1	51	22	5	11
96	0	1	1	52	23	6	12
97	0	11	11	53	24	7	1
98	0	0	0	54	25	8	2
99	0	1	1	55	27	9	3
100	0	0	112	56	27	1	4

15.5 Quellcode

Aus Platzgründen ist hier nur der wichtigste Teil unseres Quellcodes enthalten. Zusätzlich konnten wir durch Weglassen der Funktionskopf-Kommentare in DMClasses.cpp weitere 30 Seiten sparen. Den vollständigen Quellcode findet man auf der CD-ROM.