

Implementierung eines GRE Tunnels über IPSec zwischen Cisco Routern

Von Michael Thumann, mthumann@ernw.de

1 Einleitung

Dieses Papier beschreibt den Aufbau eines GRE Tunnels über IPSec, um das IPX Protokoll in ein VPN über das Internet zu integrieren.

Dieses Papier ist Teil einer Reihe von IPSec Papieren, welche als Anleitung zu IPSec Installationen verschiedenster Art dienen sollen¹.

1.1 Das Problem

Novell Netzwerke sind immer noch sehr verbreitet und speziell die älteren Versionen bauen noch auf dem IPX Protokoll auf. Eine komplette Migration eines Corporate Networks zu TCP/IP ist schwierig und zeitaufwendig, da die Verfügbarkeit der Server und des Netzes gewährleistet werden muß. Eine schleichende Migration setzt aber den Parallelbetrieb zweier Protokolle voraus.

Die zunehmende Mobilität der Mitarbeiter verlangt außerdem nach Lösungen, Mobile User und Home Offices in das Corporate Network zu integrieren. Eine Anbindung über das Internet stellt hier die kostengünstigste Lösung dar, erfordert aber einige Sicherungsmaßnahmen, um sowohl die Vertraulichkeit, Integrität sowie die Authentizität der Daten zu gewährleisten.

IPSec bietet hier Lösungsmöglichkeiten, um solche TCP/IP basierenden VPNs über das Internet aufzubauen. IPSec hat sich für diesen Einsatzzweck bereits bewährt, baut aber auf IP auf. Es enthält keine Mechanismen, um nicht TCP/IP Protokolle zu transportieren.

Hier kommt GRE (General Routing Encapsulation) von Cisco als mögliche Lösung ins Spiel. GRE ermöglicht den Transport von nicht TCP/IP Paketen über einen IP Tunnel. Dieser wiederum kann über IPSec verschlüsselt übertragen werden. Nur sehr wenige Hersteller sind in der Lage andere Protokolle über IPSec zu transportieren bzw. zu tunneln, daher fiel die Wahl auf die Komponenten der Fa. Cisco.

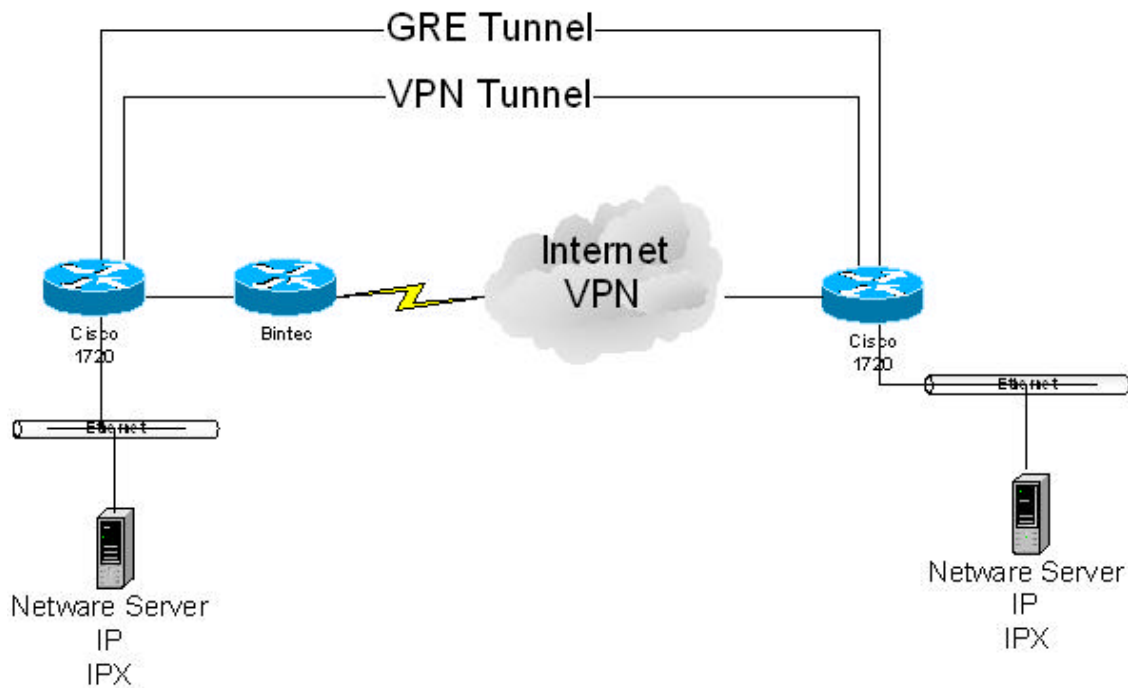
1.2 Die vorgeschlagene Lösung

Implementierung eines VPNs mit IPSec über das Internet, Einsatz von GRE als Tunnel über IP, um IPX Daten über das VPN zu übertragen. Die VPN Router (2 Cisco 1720 Router mit VPN Modul) werden in einer eigenen DMZ an die Firewall angeschlossen. Ein VPN Router steht in einer Geschäftsstelle und bekommt seine IP Adresse dynamisch zugewiesen, der zweite steht in der Hauptstelle und ist über eine fest vergebene öffentliche IP Adresse erreichbar. Es soll IP sowie IPX als Protokoll über den GRE Tunnel übertragen werden.

Für die Verschlüsselung des IPSec basierenden VPN wird Triple-DES eingesetzt, da DES heute nicht mehr als sicher betrachtet wird. Die Authentifizierung erfolgt über Preshared Secrets, dies ist zwar die unsicherste Form der Authentifizierung, aber eine CA (Certificate Authority) zur Ausgabe von Zertifikaten ist zur Zeit nicht im Einsatz.

¹ Siehe auch „Implementierung eines VPN zwischen Windows 2000 und Cisco Routern“ von Enno Rey

1.3 Das Design



1.4 Die eingesetzten Cisco Komponenten

Für diese Implementierung wurden 2 Cisco Router des Typs 1720 mit VPN Modul ausgewählt, um bei der Ver- und Entschlüsselung der Daten eine akzeptable Performance zu erzielen.

Auf diesen Router wurde das IOS (Internetwork Operating System) mit der Version 12.1(3) XT2 (ADSL / IP / IPSEC / IPX) installiert. Dieses ist ein Early Deployment Release, aber das in den 1720er Routern eingesetzte Ethernet Modul (WIC-1ENET) wird derzeit nur in Early Deployment Releases unterstützt.

Weiterhin sind zwei Novell Netware 3.12 Server in den LAN Segmenten vorhanden, die sowohl über IP wie auch IPX kommunizieren

2 Die eingesetzten Protokolle:

2.1 IPSec

IPSec ist eine standardisierte Protokollfamilie (Basis-RFC ist das RFC 2401) zur Sicherung von IP-Paketen. Sie beinhaltet Mechanismen zur Wahrung der Integrität, Vertraulichkeit und Authentizität von Traffic und zum Schutz vor wiederholtem Senden von Paketen (*anti-replay*). IPSec setzt sich aus drei Hauptkomponenten zusammen: dem Schlüsselaustausch-Verfahren *Internet Key Exchange* (IKE, RFC 2409) und den Protokollen *Encapsulation Security Payload* (ESP, RFC 2406, bietet alle o.g. Schutzmechanismen) sowie *Authentication Header* (AH, RFC 2402, bietet keine Verschlüsselung). Es können Pakete inklusive IPHeader verschlüsselt werden (man spricht dann vom *Tunnel-Modus*, der für den Gateway-zu-Gateway Einsatz, meist den Schutz kompletter Netze bei Verbindung über ein unsicheres Medium, vorgesehen ist), es kann jedoch auch nur Verkehr oberhalb Layer 3 (mit Beibehaltung einer ‚unbehandelten‘ IP-Adresse) verschlüsselt werden (dies ist der sog. *Transport-Modus*). Bei Aufnahme einer IPSec-basierten Verbindung zwischen zwei Kommunikationspartnern werden sog. *Security Associations* (SAs) hergestellt. Zunächst wird eine IKE-SA gebildet, innerhalb derer sich die Teilnehmer wechselseitig authentifizieren und sich über die für die weitere Kommunikation verwendeten Schlüssel einig werden. Anschließend werden durch IKE zwei gewissermaßen IPSec-SAs mit neuen Schlüsseln generiert (je eine für jede Richtung), die dann für den konkreten Schutz des Verkehrs (mithilfe von ESP und/oder AH) zuständig sind. Der *Internet Key Exchange* kann also in zwei Teile zerlegt werden: eine *Phase 1*, in der ein sicherer und

authentifizierter Kommunikationskanal aufgebaut wird, und eine *Phase 2*, in der dieser Kommunikationskanal genutzt wird, um die konkreten Parameter der nachfolgenden gesicherten Verbindung auszuhandeln.

Ein erfolgreicher Ablauf der Phasen setzt jeweils voraus, daß beide Seiten eine Einigung über die je anzuwendenden Parameter (etwa die Authentifizierungs-Methode oder das Verschlüsselungsverfahren) erzielt haben, die von IPsec wohlweislich nicht (kaum) vorgeschrieben sind. Die Parameter **müssen** auf beiden Seiten also übereinstimmen (können). Genau das ist die Quelle der meisten Probleme mit IPsec. In der *Phase 1* werden vor allem verhandelt:

- Authentifizierungsverfahren (z.B. *preshared keys/secret*, Signaturen mit RSA oder DSA47 [~ Zertifikate])
- Verschlüsselungsverfahren für den IKE selbst (z.B. DES, 3DES, Blowfish oder CAST-128 [RFC 2144])
- Hash-Algorithmus zur Authentifizierung von Nachrichten gemäß RFC 2104 (MD-5, SHA-1 o. Tiger)
- *Diffie-Hellman*-Gruppe (meist nur Gruppe 1 oder 2)
- Lifetime (Gültigkeitsdauer) der IKE-SA (=> des dort generierten Schlüsselmaterials) in Sekunden

Es kann nicht oft genug betont werden, wenn die Kommunikationspartner keine Übereinstimmung bzgl. dieser Parameter finden, wird ein IPsec-Verbindungsaufbau fehlschlagen.

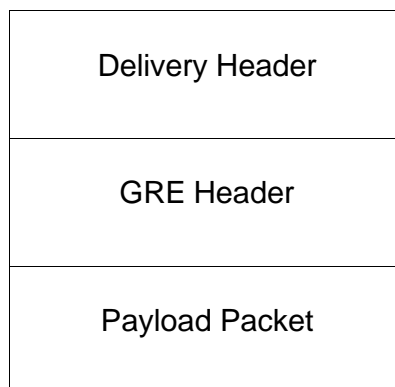
Nach erfolgreichem Abschluß der Phase 1 wird erneut verhandelt, und zwar über die Rahmenbedingungen der IPsec-SAs (die das eigentliche Ziel der Verhandlungen sind). Hier wiederum, in der *Phase 2*, sind im wesentlichen folgende Parameter (konform) zu konfigurieren:

- der Modus (*Tunnel-* oder *Transport-*Modus, siehe dazu oben)
- die Transforms (welche Sicherheitsprotokolle [ESP/AH] welche Algorithmen verwenden, Beispiele s.u.)
- die Lifetimes (Laufzeiten) d. IPsec-SAs (=>der Schlüssel), Zeit- und/oder Volumenabhängig, in Sek. o. KB
- ob *Perfect Forward Secrecy* zum Einsatz kommt, d.h. regelmäßig neues Schlüsselmaterial erzeugt wird

Erst wenn diese Einstellungen erfolgreich verhandelt sind, kommt die IPsec-Kommunikation zustande.

2.2 GRE

GRE (Generic Routing Encapsulation, RFC 1701) wurde als Protokoll entwickelt, das jedes beliebige andere Protokoll transportieren kann, indem es in GRE eingepackt (getunnelt) wird. Das eingepackte Paket hat folgenden Aufbau:



Delivery Header: Bezeichnet den Protokoll Header des Protokoll, das für den eigentlichen Transport zum Ziel zuständig ist (meistens IP)
GRE Header: Der Protokoll Header von GRE selbst
Payload Packet: Das eingepackte Protokoll, das über GRE getunnelt wird

2.3 IP und IPX

Da es zu diesen Protokollen bereits viele qualifizierte Beschreibungen gibt, sollen diese hier nicht nochmal wiederholt werden.

3 Die Implementierung

3.1 Generelle Maßnahmen

Zuerst werden beide Router für den Zugriff über SSH konfiguriert:

```
Router(config)#  
Router(config)# aaa new-model  
Router(config)# username user password geheim  
Router(config)# ip domain-name test-gre.de  
Router(config)# crypto key generate rsa 2048  
Router(config)# ip ssh time-out 60  
Router(config)# ip ssh authentication-retries 2  
Router(config)# line vty 0 4  
Router(config)# transport input ssh
```

Derzeit unterstützen Cisco Komponenten zwar lediglich SSH in der Version 1, aber die Übertragung von Benutzernamen sowie Kennwörtern erfolgt im Gegensatz zu Telnet nicht im Klartext, sondern verschlüsselt.

Zusätzlich wird ein grundsätzliches Hardening der Cisco Router durchgeführt. Das Skript basiert dabei auf einem IOS Hardening Template von Rob Thomas (robt@cymru.com):

```
service nagle  
service tcp-keepalives-in  
service tcp-keepalives-out  
! Show copious timestamps in our logs  
service timestamps debug datetime msec show-timezone localtime  
service timestamps log datetime msec show-timezone localtime  
service password-encryption  
no service dhcp  
!  
!  
logging buffered 16384 debugging  
!  
! Don't run the HTTP server.  
no ip http server  
!  
! Allow us to use the low subnet and go classless  
ip subnet-zero  
ip classless  
!  
! Disable noxious services  
no service pad  
no ip source-route  
no ip finger  
no ip bootp server  
no ip domain-lookup
```

```

!
!
!
no cdp run
!
interface Ethernet0
  no ip redirects
  no ip unreachable
  no ip directed-broadcast
  no ip proxy-arp
  no ip mask-reply
!
interface FastEthernet0
  no ip redirects
  no ip unreachable
  no ip directed-broadcast
  no ip proxy-arp
  no ip mask-reply
!
end

```

Das Hardening Skript ist noch nicht auf die besonderen Anforderungen der Aufgabenstellung angepasst, sondern stellt lediglich ein Basis Hardening dar, wie es auf fast jedem Router durchgeführt werden sollte. Es sind aber noch weitere Anpassungen erforderlich wie z. B. der Einsatz von Access Listen, welche sich aber je nach Aufgabenstellung stark unterscheiden.

3.2 Konfiguration des GRE Tunnels

Der nächste Schritt der Installation besteht im Aufbau des GRE Tunnels. Diese Vorgehensweise erscheint besonders aus Sicht des Troubleshooting ratsam, da auf diese Weise die Funktion des GRE Tunnels vor (!) dem Einsatz von IPSec geprüft werden kann. Folgende Parameter müssen auf Router A und Router B konfiguriert werden:

```

RouterA(config)# interface Tunnel0
RouterA(config)# ip address 172.16.16.1 255.255.255.252
RouterA(config)# no ip route-cache
RouterA(config)# no ip mroute-cache
RouterA(config)# tunnel source FastEthernet0
RouterA(config)# tunnel destination 10.1.0.3

```

Bei dem Interface Tunnel0 handelt es sich um ein Virtuelles Interface, es dient der Konfiguration von GRE

Auf Router B wird der zweite Tunnelendpunkt konfiguriert:

```

RouterB(config)#interface Tunnel0
RouterB(config)# ip address 172.16.16.2 255.255.255.252
RouterB(config)# no ip route-cache
RouterB(config)# no ip mroute-cache
RouterB(config)# tunnel source FastEthernet0
RouterB(config)# tunnel destination 10.0.0.3

```

Mit dem Befehl „show interface Tunnel0“ kann überprüft werden, ob der Tunnel aufgebaut wird:

```
RouterA>sh int tun0
Tunnel0 is up, line protocol is up
  Hardware is Tunnel
  Internet address is 172.16.16.1/30
  MTU 1514 bytes, BW 9 Kbit, DLY 500000 usec,
    reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation TUNNEL, loopback not set
  Keepalive set (10 sec)
  Tunnel source 10.0.0.3 (FastEthernet0), destination 10.1.0.3
  Tunnel protocol/transport GRE/IP, key disabled, sequencing disabled
  Checksumming of packets disabled, fast tunneling enabled
  Last input 00:00:01, output 00:00:01, output hang never
  Last clearing of "show interface" counters never
  Queueing strategy: fifo
  Output queue 0/0, 0 drops; input queue 0/75, 0 drops
  5 minute input rate 0 bits/sec, 1 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
    16727 packets input, 2290797 bytes, 0 no buffer
    Received 0 broadcasts, 0 runts, 0 giants, 0 throttles
    0 input errors, 0 CRC, 0 frame, 0 overrun, 0 ignored, 0 abort
    26900 packets output, 14081762 bytes, 0 underruns
    0 output errors, 0 collisions, 0 interface resets
    0 output buffer failures, 0 output buffers swapped out
```

3.3 Konfiguration von IPX

Nun wird das IPX Routing konfiguriert:

```
RouterA(config)# ipx routing
RouterB(config)# ipx routing
```

Und natürlich die IPX Netze. Zuerst das Transit Netz:

```
RouterA(config)# interface Tunnel0
RouterA(config)# ipx network AABBC
```

```
RouterB(config)# interface Tunnel0
RouterB(config)# ipx network AABBC
```

Dann die lokalen IPX Netze sowie der Ethernet Frame Typ:

```
RouterA(config)# interface FastEthernet0
RouterA(config)# ipx encapsulation ARPA
RouterA(config)# ipx network F110E2
```

```
RouterB(config)# interface FastEthernet0
RouterB(config)# ipx encapsulation ARPA
RouterB(config)# ipx network F101E2
```

Mit zwei „show“- Befehlen läßt sich jetzt prüfen, ob sich die beiden IPX Netze kennen bzw. die jeweiligen Netware Server sichtbar sind:

```
RouterA#show ipx route
Codes: C - Connected primary network,      c - Connected secondary network
       S - Static, F - Floating static, L - Local (internal), W - IPXWAN
       R - RIP, E - EIGRP, N - NLSP, X - External, A - Aggregate
```

s - seconds, u - uses, U - Per-user static

10 Total IPX routes. Up to 1 parallel paths and 16 hops allowed.

No default route known.

```
C      AABBC (TUNNEL),      Tu0
C      F110E2 (ARPA),      Fa0
R      F101E2 [151/01] via  AABBC.0004.dd0c.0c10,  23s, Tu0
R      38AFB23 [02/01] via  F110E2.0060.08f6.a51a,  37s, Fa0
R      4071998 [02/01] via  F110E2.0010.5ad5.b736,  27s, Fa0
R      71121E8 [02/01] via  F110E2.0060.08f6.ce97,  13s, Fa0
R      9071998 [152/02] via  AABBC.0004.dd0c.0c10,  23s, Tu0
R      AF594592 [03/01] via  F110E2.0010.5ad5.b736,  27s, Fa0
R      F1018022 [152/02] via  AABBC.0004.dd0c.0c10,  23s, Tu0
R      F1108022 [02/01] via  F110E2.0060.08f6.ce97,  13s, Fa0
```

RouterA#show ipx server

Codes: S - Static, P - Periodic, E - EIGRP, N - NLSP, H - Holddown, + = detail

U - Per-user static

5 Total IPX Servers

Table ordering is based on routing and server info

Type	Name	Net	Address	Port	Route	Hops	Itf
P	4 ServerVERW	71121E8.0000.0000.0001	0000.0001:0451		2/01	1	Fa0
P	4 ServerPROD	9071998.0000.0000.0001	0000.0001:0451		152/02	2	Tu0
P	47 PS_Server_PROD	9071998.0000.0000.0001	0000.0001:8060		152/02	3	Tu0
P	107 ServerVERW	71121E8.0000.0000.0001	0000.0001:8104		2/01	1	Fa0
P	107 Server_PROD	9071998.0000.0000.0001	0000.0001:8104		152/02	3	Tu0

Beachten Sie, über welche Interfaces die Netze bzw. auch die Server erkannt werden. Tu0 steht dabei für das Tunnel Interface 0, also in diesem Fall das über diesen Tunnel angeschlossene Netz. Die IPX Verbindung zwischen den beiden Netzen funktioniert also.

3.4 Konfiguration von IPSec

Nachdem die grundlegende Funktion hergestellt ist, sollen die Daten durch Verschlüsselung geschützt werden. Der Aufbau des VPNs über IPSec ist also der nächste Installationsschritt.

Vorbereitend werden als erstes die Kommunikationsparameter festgelegt, um von vornherein Parameter Probleme zu vermeiden:

Phase 1 :

- Authentifizierungsverfahren: *Preshared Keys/Secret*
- Verschlüsselungsverfahren für den IKE selbst: 3DES
- Hash-Algorithmus zur Authentifizierung von Nachrichten gemäß RFC 2104: SHA-1
- *Diffie-Hellman-Gruppe: 2*
- Lifetime (Gültigkeitsdauer) der IKE-SA in Sekunden: 86400 (Ciscos Default Einstellung)

Phase 2 :

- der Modus: *Tunnel- Modus*
- die Transforms: ESP und sha-hmac
- die Lifetimes (Laufzeiten) d. IPsec-SAs in Sekunden: 86400 (Ciscos Default Einstellung)
- *Perfect Forward Secrecy: Ja, mit DH Gruppe 2*

Zuerst werden auf den Cisco Routern die ISAKMP Parameter konfiguriert:

```
RouterA(config)# crypto isakmp policy 1
RouterA(config)# encryption 3des
RouterA(config)# hash sha
RouterA(config)# authentication pre-share
RouterA(config)# group 2
```

```
RouterB(config)# crypto isakmp policy 1
RouterB(config)# encryption 3des
RouterB(config)# hash sha
RouterB(config)# authentication pre-share
RouterB(config)# group 2
```

Hier ist es besonders wichtig, den Schlüsselaustausch mit dem folgenden Befehl einzuschalten, da der Schlüsselaustausch sonst niemals initiiert wird:

```
RouterA(config)# crypto isakmp enable
RouterB(config)# crypto isakmp enable
```

Nun wird der Preshared Key auf beiden Router festgelegt. Da sich der Router in der Aussenstelle mit wechselnden IP Adresse beim Hauptstellen Router melden wird müssen die Peers hier unterschiedlich konfiguriert werden, damit sich die beiden Router auch gegenseitig als IPSec Kommunikationspartner akzeptieren:

```
RouterA(config)#crypto isakmp key **** address 0.0.0.0 0.0.0.0
RouterB(config)# crypto isakmp key **** address xxx.xxx.xxx.xxx
```

Die Peer Adresse 0.0.0.0 bedeutet, daß der Router A jeden anderen Router, der sich erfolgreich mittels des Preshared Keys authentifiziert, als Kommunikationspartner zulassen wird.

Dann wird die Verkehrsverschlüsselung konfiguriert:

Hier sind verschiedene Schritte notwendig. Zunächst muß definiert werden, *welcher* Verkehr überhaupt verschlüsselt werden soll. Dann wird durch ein *transform* festgelegt, *wie* (d.h. mit welchen Verschlüsselungs- und/oder Hash-Algorithmen) diese Pakete geschützt werden sollen. Und schließlich werden diese Einstellungen unter dem Dach einer *crypto map* zusammengefasst, die dann wiederum auf ein Interface angewendet wird.

Auf welchen Verkehr IPsec angewandt wird, regelt eine *crypto access-list*. *Crypto access-lists* sehen syntaktisch aus wie reguläre *access-lists*, haben aber eine gänzlich andere Funktion: alle Pakete, die darin unter ein *permit*-Statement fallen, sollen verschlüsselt werden.

Initiale IPsec-Pakete (IKE), die den Schlüsselaustausch durchführen (ISAKMP-Verkehr, UDP Port 500), werden unverschlüsselt zugelassen. Sonstiger Verkehr wird gewissermaßen IPsec unterworfen (mittels eines *permit*-Statements!). Dies betrifft insbesondere auch GRE und das dort eingepackte IPX.

Router A:

```
access-list 101 deny    udp any any eq isakmp
access-list 101 permit gre host 10.0.0.3 host 10.1.0.3
access-list 101 permit ip 10.0.0.0 0.0.0.255 10.1.0.0 0.0.0.255
```

Router B:

```
access-list 101 deny    udp any any eq isakmp
access-list 101 permit gre host 10.1.0.3 host 10.0.0.3
access-list 101 permit ip 10.1.0.0 0.0.0.255 10.0.0.0 0.0.0.255
```

Achten Sie peinlichst auf korrekte *access-lists*, auch diese stellen eine häufige Fehlerquelle bei IPSec Implementationen dar!

Anschließend wird mittels einer Transformation (eines *transforms*) definiert, welche Sicherheitsprotokolle mit welchen Algorithmen zur (Verhandlungs-) Auswahl stehen. Wir

beschränken uns auf ESP mit 3DES als Verschlüsselungs- sowie SHA-1 als Hash-Algorithmus und arbeiten im *tunnel mode*.

```
RouterA(config)# crypto ipsec transform-set gretunnel esp-3des esp-sha-hmac
RouterA(config)# mode tunnel
```

```
RouterB(config)# crypto ipsec transform-set gretunnel esp-3des esp-sha-hmac
RouterB(config)# mode tunnel
```

Die *crypto access-list* und der *transform* werden jetzt verknüpft durch eine *crypto map*. Eine solche *crypto map* regelt, welcher Verkehr geschützt wird (in Form der *access-list*), an welche Gegenstelle dieser Traffic geschickt wird (in Form der Angabe eines o. mehrerer *peers*), welche Sicherheits-Protokolle mit welchen Algorithmen dafür akzeptabel sind (via *transform*) und ggf. wie Schlüssel bzw. SAs verwaltet werden.

Sind diese Informationen nicht im voraus bekannt (wie im Fall der dynamischen IP-Adresse des anderen Routers), kommt eine *dynamic map* in's Spiel. Dabei handelt es sich um eine Art Template, das variabel angewendet werden kann.

Erstellt wird auf Router B eine *dynamic map* namens **gre-dy** mit verschiedenen Parametern (*transform*, *access-list*, PFS mit DH-Gruppe 2 aktiviert) und eine *crypto map* namens **gre**, die auf das dynamische Template ‚gre‘ zurückgreift. Da Router B aber die fest vergebene IP Adresse von Router A kennt (xx.xx.xx.xx) ist hier die Konfiguration einer *dynamic map* nicht notwendig.

Router A:

```
RouterA(config)# crypto dynamic-map gre-dy 1
RouterA(config)# set transform-set gretunnel
RouterA(config)# set pfs group2
RouterA(config)# match address 101
RouterA(config)# exit
RouterA(config)# crypto map gre 1 ipsec-isakmp dynamic gre-dy
```

Router B:

```
RouterB(config)# crypto map gre 1 ipsec-isakmp
RouterB(config)# set peer xxx.xxx.xxx.xxx
RouterB(config)# set transform-set gretunnel
RouterB(config)# set pfs group2
RouterB(config)# match address 101
```

Diese *crypto map* wird dann an einem Interface angewendet:

Router A:

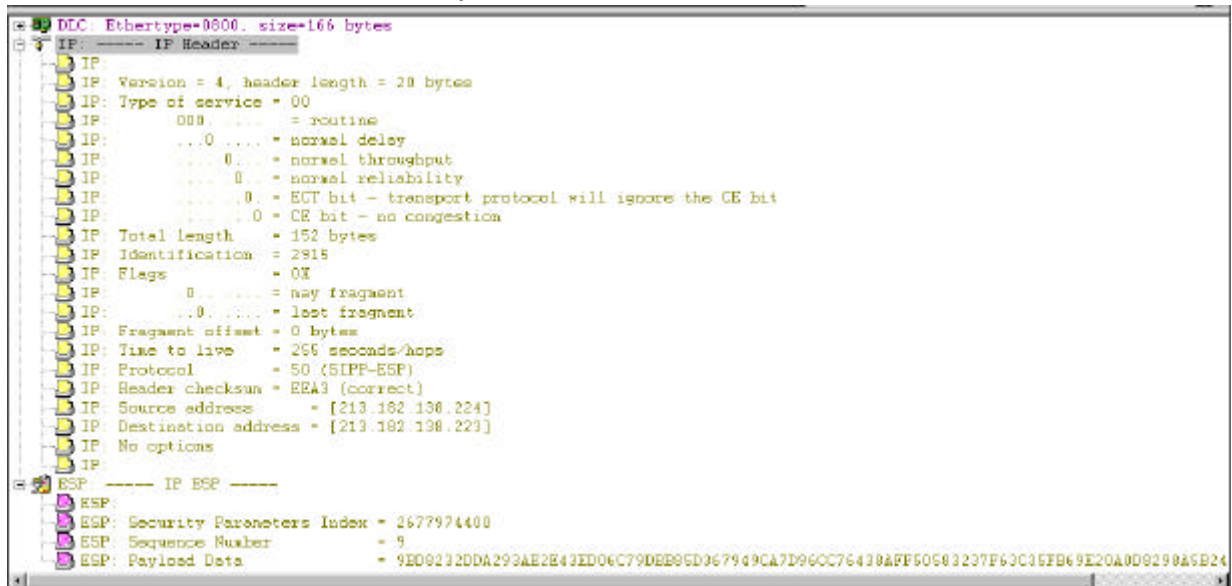
```
RouterA(config)# interface Ethernet0
RouterA(config)# crypto map gre
```

Router B:

```
RouterB(config)# interface Ethernet0
RouterB(config)# crypto map gre
```

Das VPN sollte jetzt funktionieren und sowohl IP wie auch über GRE getunneltes IPX verschlüsselt übertragen.

Ein Blick mit einem Protokoll Analyzer verifiziert das Ganze:



4 Zum Schluß

4.1 Danksagungen

Enno Rey (erey@ernw.de) für seine Unterstützung und seine IPSec Beiträge.

4.2 Disclaimer

Alle im Dokument genannten Produkte sind Warenzeichen der jeweiligen Hersteller. Ich übernehme keinerlei Haftung oder Garantie für das Gelingen der beschriebenen Implementierung oder Folgen derselben oder eines Fehlschlags. Halt einfach keine Gewährleistung irgendeiner Art.

Über Berichtigungen, Hinweise, Kommentare, Feedback gleich welcher Art bin ich immer erfreut.

4.3 OpenContent License

Das Papier steht unter der OpenContent License (www.opencontent.org/opl.shtml) und kann entsprechend verteilt, ergänzt, zitiert, etc. werden.

Anhang A

Und hier nochmal die kompletten Router Konfigurationen:

Router A:

```
Current configuration : 4073 bytes
!
version 12.1
no service single-slot-reload-enable
service nagle
no service pad
service tcp-keepalives-in
service tcp-keepalives-out
service timestamps debug datetime msec localtime show-timezone
service timestamps log datetime msec localtime show-timezone
service password-encryption
```

```
no service dhcp
!
hostname gre-main
!
logging buffered 16384
logging rate-limit console 10 except errors
aaa new-model
enable secret 5 *****
!
username user password 7 *****
memory-size iomem 20
clock timezone berlin 1
clock summer-time berlin recurring
ip subnet-zero
no ip source-route
no ip finger
no ip domain-lookup
ip domain-name kuo.de
!
no ip bootp server
ip audit notify log
ip audit po max-events 100
ip ssh time-out 120
ip ssh authentication-retries 3
ipx routing 0004.dd0c.0aec
!
!
crypto isakmp policy 1
  encr 3des
  authentication pre-share
  group 2
crypto isakmp key **** address 0.0.0.0 0.0.0.0
!
!
crypto ipsec transform-set gretunnel esp-3des esp-sha-hmac
!
crypto dynamic-map gre-dy 1
  set transform-set gretunnel
  set pfs group2
  match address 101
!
!
crypto map gre 1 ipsec-isakmp dynamic gre-dy
!
!
!
!
interface Tunnel0
  ip address 172.16.16.1 255.255.255.252
  no ip route-cache
  no ip mroute-cache
  ipx network AABCC
  tunnel source FastEthernet0
  tunnel destination 10.1.0.3
  crypto map gre
!
interface Ethernet0
```

```

ip address xxx.xxx.xxx.xxx 255.255.255.0
ip access-group 107 in
no ip redirects
no ip unreachableables
no ip proxy-arp
half-duplex
no cdp enable
crypto map gre
!
interface FastEthernet0
description connected to EthernetLAN
ip address 10.0.0.3 255.255.255.0
no ip redirects
no ip unreachableables
no ip proxy-arp
speed auto
ipx encapsulation ARPA
ipx network F110E2
no cdp enable
!
ip kerberos source-interface FastEthernet0
ip classless
ip route 0.0.0.0 0.0.0.0 xxx.xxx.xxx.xxx
no ip http server
!
access-list 101 deny    udp any any eq isakmp
access-list 101 deny    udp any any eq domain
access-list 101 permit  gre host 10.0.0.3 host 10.1.0.3
access-list 101 permit  ip 10.0.0.0 0.0.0.255 10.1.0.0 0.0.255.255
no cdp run
!
!
!
!
line con 0
transport input none
line aux 0
line vty 0 4
password 7 *****
transport input ssh
!
no scheduler allocate
end

```

Router B:

```

Current configuration : 2220 bytes
!
version 12.1
no service single-slot-reload-enable
service nagle
no service pad
service tcp-keepalives-in
service tcp-keepalives-out
service timestamps debug datetime msec localtime show-timezone

```

```
service timestamps log datetime msec localtime show-timezone
service password-encryption
no service dhcp
!
hostname gre-bo
!
logging buffered 16384
logging rate-limit console 10 except errors
aaa new-model
enable secret 5 *****
!
username user password 7 *****
memory-size iomem 20
clock timezone berlin 1
clock summer-time berlin recurring
ip subnet-zero
no ip source-route
no ip finger
no ip domain-lookup
ip domain-name kuo.de
!
no ip bootp server
ip audit notify log
ip audit po max-events 100
ip ssh time-out 120
ip ssh authentication-retries 3
ipx routing 0004.dd0c.0c10
!
!
crypto isakmp policy 1
  encr 3des
  authentication pre-share
  group 2
crypto isakmp key **** address xxx.xxx.xxx.xxx
!
!
crypto ipsec transform-set gretunnel esp-3des esp-sha-hmac
!
crypto map gre 1 ipsec-isakmp
  set peer xxx.xxx.xxx.xxx
  set transform-set gretunnel
  set pfs group2
  match address 101
!
!
!
!
interface Tunnel0
  ip address 172.16.16.2 255.255.255.252
  no ip route-cache
  no ip mroute-cache
  ipx network AABCC
  tunnel source FastEthernet0
  tunnel destination 10.0.0.3
  crypto map gre
!
interface BRI0
```

```
no ip address
shutdown
no cdp enable
!
interface Ethernet0
 ip address 192.168.169.2 255.255.255.0
 no ip redirects
 no ip unreachable
 no ip proxy-arp
 half-duplex
 no cdp enable
 crypto map gre
!
interface FastEthernet0
 description connected to EthernetLAN
 ip address 10.1.0.3 255.255.255.0
 no ip redirects
 no ip unreachable
 no ip proxy-arp
 speed auto
 ipx encapsulation ARPA
 ipx network F101E2
 no cdp enable
!
ip kerberos source-interface FastEthernet0
ip classless
ip route 0.0.0.0 0.0.0.0 192.168.169.1
no ip http server
!
access-list 101 deny    udp any any eq isakmp
access-list 101 deny    udp any any eq domain
access-list 101 permit gre host 10.1.0.3 host 10.0.0.3
access-list 101 permit ip 10.1.0.0 0.0.0.255 10.0.0.0 0.0.255.255
no cdp run
!
!
!
!
line con 0
  transport input none
line aux 0
line vty 0 4
  transport input ssh
!
end
```