IPsec mit dem Linux Kernel 2.6

©2003 Ralf Spenneberg*

$Revision$: 1.4

Zusammenfassung

Linux unterstützt seit einigen Jahren virtuelle private IPv4 Netzwerke mit IPsec durch das FreeS/WAN Projekt. Verschiedene Gründe (Lizenz, Codequalität, u.a.) haben jedoch bisher eine Aufnahme in den Linux Kernel verhindert. Die selben Gründe sind dafür verantwortlich, dass FreeS/WAN von einigen Distributionen (z.B. Red Hat) nicht vertrieben wird.

Seit dem Entwicklerkernel 2.5.45 existiert eine neue IPsec Implementierung fest in dem Linux Kernel, die auf dem USAGI Projekt basiert. Diese wurde von Dave Miller und Alexey Kuznetsov in den Kernel portiert.

1 Warum ?

Verschiedene Personen haben mehrfach die Entwicklung einer alternativen IPsec Implementierung oder einer Reimplementierung von FreeS/WAN gefordert. Dies hatte mehrere Gründe. So wurde von dem FreeS/WAN Begründer John Gilmore 1996 verfügt, dass kein US Amerikaner an dem FreeS/WAN Projekt mitarbeiten dürfe. Hiermit sollte das Projekt, unabhängig von den damals geltenden US Exportbestimmungen, starke Kryptographie entwickeln können. Diese Einschränkung ist von dem FreeS/WAN Projekt bis heute nicht aufgehoben worden, obwohl die Exportbestimmungen der USA modifiziert wurden. Zusätzlich äußerten einige Personen starke Bedenken an der Qualität des Kernel Patches KLIPS. Dieser Anteil von FreeS/WAN soll bereits seit einigen Jahren überarbeitet werden. Aus Anlass der alternativen IPsec Implementierung im Linux Kernel 2.5 schrieb Henry Spencer, ein ehemaliger führender Entwickler des FreeS/WAN Projektes, am 8. November 2002 in einer E-Mail: *And, in fairness, KLIPS is the ugliest and least maintainable part of FreeS/WAN, and deserves to be supplanted.*

Diese Gründe führten zu der Implementierung von IPsec für IPv4 und IPv6 in dem Linux Kernel 2.5.45 durch Dave Miller und Alexey Kuznetsov. Hierbei basierte ihre Arbeit im wesentlichen auf dem USAGI Projekt[1], dem KAME Projekt[2] und dem WIDE Projekt[3].

Diese Implementierung wird jedoch sicherlich nicht zum Ende des FreeS/WAN Projektes führen. Auf der FreeS/WAN-Mailingliste wurde bereits mehrfach die Portierung der FreeS/WAN Werkzeuge auf die neue Kernelimplementierung besprochen und auch angekündigt. Hiermit ist bei Erscheinen des Linux Kernels 2.6 zu rechnen, so dass dann vorhandene und funktionsfähige VPN Netzwerke auf den neuen Linux Kernel ohne große Anpassungen migriert werden können.

*ralf@spenneberg.net

[1] http://www.linux-ipv6.org

[2] http://www.kame.net

[3] http://www.wide.ad.jp

2 Was ?

Die IPsec Implementierung ist seit dem Linux Kernel 2.5.47 funktionsfähig und relativ komplett. So können IPsec Verbindungen mit dem Authentication Header Protokoll (AH) und dem Encapsulating Security Payload Protokoll (ESP) aufgebaut werden. Diese Verbindungen können in dem so genannten Transport oder Tunnel Mode erfolgen.

IPsec bietet zwei verschiedene Protokolle für den Aufbau von gesicherten Verbindungen:

- Authentication Header (AH) IP Protokoll 51. Dieses Protokoll versieht jedes Paket mit einem Message Authentication Code (MAC). Hierbei handelt es sich um eine kryptographische Prüfsumme, die aus dem Inhalt des IP Paketes und einem geheimen Schlüssel ermittelt wird. Der geheime Schlüssel ist nur den Kommunikationspartnern bekannt. Daher können nur diese den MAC erzeugen und überprüfen. Modifikationen des Paketes können so sofort und einwandfrei festgestellt werden. Dabei schützt das AH Protokoll das gesamte IP Paket einschließlich der unveränderlichen Informationen des IP Headers. Dies sind unter anderem auch die Absender und die Empfänger IP Adresse. Der 96 Bit lange MAC wird üblicherweise mit dem Algorithmus MD5 oder SHA1 erzeugt. Moderne Implementierungen erlauben auch den Einsatz von SHA-2.

- Encapsulating Security Payload (ESP) IP Protokoll 50. Dieses Protokoll ist in der Lage den Inhalt eines Paketes sowohl zu verschlüsseln, als auch mit einem MAC gegen Modifikationen zu schützen. Der MAC des ESP Protokolls hat dieselbe Aufgabe wie der MAC des AH Protokolls. Jedoch bezieht sich dieser MAC lediglich auf den Inhalt des IP Paketes und betrifft nicht den IP Header. Die wesentliche Aufgabe des ESP Protokolls ist jedoch die Verschlüsselung. Hierzu werden die Daten mit einem symmetrischen Verschlüsselungsalgorithmus und einem geheimen vorher ermittelten Schlüssel verschlüsselt. Übliche Verschlüsselungsalgorithmen sind NULL, DES und 3DES. In letzter Zeit werden häufig auch AES, Blowfish, Twofish und CAST unterstützt.

Im Falle der neuen Linux IPsec Implementierung nutzt der Kernelcode die neue CryptoAPI. Daher werden die wichtigsten Algorithmen zur Authentifizierung und Verschlüsselung unterstützt. Neue Algorithmen können in der Zukunft leicht hinzugefügt werden.

Die IPsec Protokolle können sowohl im Transport Mode als auch im Tunnel Mode eingesetzt werden. Im Transport Mode verwenden die wahren Kommunikationspartner das AH und das ESP Protokoll um den Austausch der IP Pakete zu sichern. Hierbei erhält das IP Paket einen zusätzlichen AH und/oder ESP Header bevor es seinen IP Header erhält. Der Transport Mode kann immer nur zwischen zwei Rechnern genutzt werden.

Der Tunnel Mode wird wesentlich häufiger eingesetzt. Hierbei werden die kompletten IP Pakete erneut mit dem AH und/oder dem ESP Protokoll verpackt. Dabei werden die Pakete auch mit dem ESP Protokoll verschlüsselt. Anschließend erhalten die Pakete einen neuen äußeren IP Header. Dieser neue IP Header steuert dann den Transport des Paketes zum anderen Endpunkt des Tunnels. Dort wird das Paket authentifiziert und entschlüsselt. Hierbei wird das originale Paket wiederhergestellt. Dieser Modus kann zwischen zwei einzelnen Rechner genutzt werden, jedoch wird er wesentlich häufiger für den Aufbau verschlüsselter VPN Verbindungen zwischen zwei Netzwerken genutzt. Hierzu werden zwei VPN Gateways im Tunnel Mode betrieben, die alle Pakete, die von

dem einen Netzwerk in das andere Netzwerk gelangen möchten, für den Transport authentifizieren und verschlüsseln.

3 Wie ?

Für die folgenden Demonstrationen wird ein aktueller Kernel $>= 2.5.47$ benötigt. Dieser Kernel muss mindestens mit den folgenden Optionen konfiguriert worden sein:

```
Networking support (NET) [Y/n/?] y
  *
  * Networking options
  *
  PF_KEY sockets (NET_KEY) [Y/n/m/?] y
  IP: AH transformation (INET_AH) [Y/n/m/?] y
  IP: ESP transformation (INET_ESP) [Y/n/m/?] y
  IP: IPsec user configuration interface (XFRM_USER) [Y/n/m/?] y

Cryptographic API (CRYPTO) [Y/n/?] y
  HMAC support (CRYPTO_HMAC) [Y/n/?] y
  Null algorithms (CRYPTO_NULL) [Y/n/m/?] y
  MD5 digest algorithm (CRYPTO_MD5) [Y/n/m/?] y
  SHA1 digest algorithm (CRYPTO_SHA1) [Y/n/m/?] y
  DES and Triple DES EDE cipher algorithms (CRYPTO_DES) [Y/n/m/?] y
  AES cipher algorithms (CRYPTO_AES) [Y/n/m/?] y
```

Wenn nun mit IPsec verschlüsselte Verbindungen aufgebaut werden sollen, so existieren grundsätzlich zwei verschiedene Möglichkeiten zur Vorgehensweise. Diese unterscheiden sich in dem Austausch der zu verwendenden Schlüssel. Für die Authentifizierung und die Verschlüsselung der Pakete werden symmetrische Verfahren eingesetzt. Daher benötigen beide Kommunikationspartner die identischen Schlüssel. Entweder der Austausch dieser Schlüssel erfolgt von Hand und man spricht von einer manuell verschlüsselten Verbindung, oder der Austausch erfolgt automatisch. Hierfür wurde das Internet Key Exchange (IKE) Protokoll entwickelt. Dieses Protokoll ermöglicht nach Authentifizierung des Partners einen sicheren Schlüsselaustausch nach Diffie und Hellmann. Hierbei können die Schlüssel über einen unverschlüsselten Kanal so ausgetauscht werden, dass eine dritte Person keinerlei Zugang erhält. Das IKE Protokoll sorgt anschließend auch für einen regelmäßigen Wechsel dieser Schlüssel um ihre Sicherheit zu gewährleisten.

Der Linux Kernel ist nicht in der Lage das IKE Protokoll zu verwenden. Hierfür ist (wie bei FreeS/WAN auch) ein Userspace Daemon verantwortlich. Momentan wird der Racoon IKE Daemon des KAME Projektes propagiert. Später ist sicherlich auch der Einsatz des FreeS/WAN Pluto IKE Daemons möglich.

Im folgenden wird sowohl der Aufbau einer manuell verschlüsselten Verbindung als auch einer automatisch verschlüsselten Verbindung beschrieben.

3.1 Konfiguration manuell verschlüsselter Verbindungen

Im Gegensatz zu FreeS/WAN implementiert der Linux Kernel $>= 2.5.47$ eine echte Security Association Database (SAD) und eine Security Policy Database (SPD). Ein

direkter Zugriff und eine direkte Manipulation dieser Datenbanken ist mit dem Kommando `setkey` möglich.

Die SAD enthält die IPsec Security Associations. Diese definieren, wie die IP Pakete mit IPsec zu behandeln sind. Zur eindeutigen Zuordnung sind bei einer SA folgende Angaben erforderlich:

- Quell- und Ziel-IP
- IPsec Protokoll
- Security Parameter Index (SPI)
- Algorithmus
- Geheimer Schlüssel

Die IPsec SAs müssen auf den beiden Rechnern, die miteinander kommunizieren wollen, identisch konfiguriert sein. Nur wenn beide Rechner identische Schlüssel, Algorithmen und SPIs verwenden, ist eine Kommunikation möglich.

Die SPD enthält die Security Policies. Diese definieren, wann und wie die SAs anzuwenden sind. Eine Security Policy benötigt die folgenden Informationen:

- IP Adressen der kommunizierenden Rechner
- Tunnel oder Transport Mode
- Zu verschlüsselndes Protokoll (und Port)
- Anzuwendendes Protokoll (ESP oder AH)
- Richtung der Policy

Zur Konfiguration dieser Informationen wird der Befehl `setkey` verwendet. Dieser liest die Konfiguration aus einer Datei und trägt sie in den Datenbanken des Kernels ein.

```
#!/usr/sbin/setkey -f

# Lösche die SAD und SPD
flush;
spdflush;

# manuelle Parameter für AH SAs mit 128 Bit Schlüssel für MD5
add 192.168.1.5 192.168.2.5 ah 0x300 -A hmac-md5 \
 0xdf84cd88405b8faed89031e4118e6cf6;
add 192.168.2.5 192.168.1.5 ah 0x400 -A hmac-md5 \
 0xdf84cd88405b8faed89031e4118e6cf6;

# manuelle Parameter für ESP SAs mit 192 Bit Schlüssel für 3DES (168 + 24)
add 192.168.1.5 192.168.2.5 esp 0x301 -E 3des-cbc \
 0x78bab1a6380fa764e4be09da2e13d65076d503cf3089ea82;
add 192.168.2.5 192.168.1.5 esp 0x401 -E 3des-cbc \
 0x78bab1a6380fa764e4be09da2e13d65076d503cf3089ea82;
```

```
# Richtlinien zur Verwendung der SAs
spdadd 192.168.1.5 192.168.2.5 any -P out ipsec
           esp/transport//require
           ah/transport//require;

spdadd 192.168.2.5 192.168.1.5 any -P in ipsec
           esp/transport//require
           ah/transport//require;
```

Wird nun ein Ping zwischen 192.168.1.5 und 192.168.2.5 gestartet, so zeichnet `tcpdump` folgenden Verkehr auf:

```
12:45:39.373005 192.168.1.5 > 192.168.2.5: AH(spi=0x00000300,seq=0x1):
ESP(spi=0x00000301,seq=0x1) (DF)
12:45:39.448636 192.168.2.5 > 192.168.1.5: AH(spi=0x00000400,seq=0x1):
ESP(spi=0x00000401,seq=0x1)12:45:40.542430
```

So können mit dem Kommando `setkey` sehr einfach manuell verschlüsselte Verbindungen aufgebaut werden. Diese sind in allen Punkten interoperabel mit den entsprechenden manuell verschlüsselten Verbindungen von FreeS/WAN. Manuell verschlüsselte Verbindungen werden jedoch in der Realität selten genutzt, da sie statische Schlüssel verwenden und nur über eine geringe Sicherheit verfügen. Üblicherweise verwendet man daher automatisch verschlüsselte Verbindungen. Hierbei wird von einem IKE Daemon der Schlüsselaustausch vorgenommen.

3.2 Konfiguration automatisch verschlüsselter Verbindungen mit Racoon

Bei automatisch verschlüsselten Verbindungen werden die zu verwendenden Schlüssel nicht von dem Administrator manuell vorgegeben, sondern mit dem IKE Protokoll ausgehandelt. Hierfür authentifiziert zunächst der IKE Daemon den Kommunikationspartner. Übliche Verfahren für diese Authentifizierung basieren auf Preshared Keys (PSK), RSA Schlüsseln, PGP Schlüsseln oder x509 Zertifikaten. Wurde diese Authentifizierung erfolgreich durchgeführt, so wird eine Internet Security Association Key Management Protocol Security Association (ISAKMP SA) erzeugt. Diese bidirektionale SA steuert den Austausch der Sitzungsschlüssel für die Authentifizierung und Verschlüsselung der IP Pakete. Auf der Basis dieser ISAKMP SA einigen sich die Kommunikationspartner auf die zu verwendenden Algorithmen zur Verschlüsselung und Authentifizierung der Pakete, die Verfahren zur Erzeugung der Schlüssel und deren Lebensdauer.

Das IKE Protokoll wird üblicherweise nicht von dem Betriebssystem selbst umgesetzt, sondern von einem zusätzlichen IKE Daemon verwendet. Klassische IKE Daemonen sind Pluto (FreeS/WAN), isakmpd (OpenBSD) und racoon (KAME-Projekt für *BSD). Für die hier besprochene Linux Implementierung können der racoon IKE Daemon und der isakmpd (mit einem Patch http://www.thinknerd.de/~thomas/IPsec/) eingesetzt werden. Im folgenden soll jedoch die Implementierung mit dem racoon IKE Daemon vorgestellt werden.

Der Racoon IKE Daemon bietet die Unterstützung von:

- Phase 1 des IKE Protokolls im Main, Aggressive und Base Mode
- Authentifizierung mit PSK, x509 Zertifikaten, Kerberos

- Verschlüsselung mit Null, DES, 3DES, AES, Blowfish, Twofish, CAST ...
- Authentifizierung mit MD5, SHA1, SHA2-256, SHA2-385 und SHA2-512
- Diffie Hellmann Gruppen modp768, modp1024, ec2n155, ec2n185, modp1536, modp2048, modp3072, modp4096, modp6144 und modp8192
- Perfect Forward Secrecy (PFS)
- Unterstützung für Roadwarrior

Die Konfiguration von Racoon ist recht einfach. Im folgenden ist die Konfigurationsdatei von Racoon für die Authentifizierung mit x509 Zertifikaten demonstriert:

```
path certificate "/etc/racoon/certs";

remote 192.168.2.5 {
        exchange_mode main;
        certificate_type x509 "left_cert.pem" "right_req.pem";
        my_identifier asn1dn;
peers_identifier asn1dn;
        proposal {
                encryption_algorithm 3des;
                hash_algorithm md5;
                authentication_method rsasig;
                dh_group modp1024;
        }
}

sainfo address 10.0.2.0/24 any address 10.0.1.0/24 any {
        pfs_group modp768;
        encryption_algorithm 3des;
        authentication_algorithm hmac_md5;
        compression_algorithm deflate;
}
```

Diese Konfigurationsdatei weist Racoon an, einen Tunnel von 10.0.2.0/24 nach 10.0.1.0/24 aufzubauen. Hierzu soll die Authentifizierung mit Zertifikaten erfolgen, die automatisch von den Kommunikationspartnern mit dem IKE Protokoll ausgetauscht werden. Die Identität der Kommunikationspartner wird aus dem Zertifikat in der ASN.1 Notation ausgelesen. Die Zertifikate, privaten Schlüssel und das Zertifikat der CA befinden sich in dem Verzeichnis `/etc/racoon/certs`.

Damit Racoon in der Lage ist die Zertifikate zu überprüfen, benötigt es das Zertifikat der CA, die die Zertifikate der VPN Kommunikationspartner unterzeichnet hat. Dieses Zertifikat muss einen kodierten Namen besitzen, damit es gefunden wird. Die Erzeugung dieses Namens erfolgt am einfachsten mit:

```
ln -s cacert.pem ‘openssl x509 -noout -hash -in cacert.pem ‘.0
```

Nun ist der Aufbau des Tunnels mit Racoon möglich. Racoon baut jedoch den Tunnel nicht auf eigene Veranlassung auf. Hierfür muss Racoon von dem Linux Kernel dazu aufgefordert werden. Befinden sich in der SPD Security Policies, für die keine

entsprechenden SAs in der SAD existieren, so ruft der Linux Kernel den IKE Daemon auf und beauftragt ihn die entsprechenden SAs zu erzeugen.

Hier werden die folgenden Einträge benötigt:

```
#!/usr/sbin/setkey -f
flush;
spdflush;

# Richtlinien zur Verwendung der SAs (Tunnel LeftNet-RightNet)
spdadd 10.0.1.0/24 10.0.2.0/24 any -P out ipsec
           esp/tunnel/192.168.1.5-192.168.2.5/require;

spdadd 10.0.2.0/24 10.0.1.0/24 any -P in ipsec
           esp/tunnel/192.168.2.5-192.168.1.5/require;
```

Wird nun der Tunnel mit einem Ping gestartet, so erfolgt automatisch der Aufbau:

```
2003-01-21 21:11:17: INFO: main.c:170:main(): @(#)racoon 20001216 20001216
sakane@kame.net
2003-01-21 21:11:17: INFO: main.c:171:main(): @(#)This product linked OpenSSL
0.9.6b [engine] 9 Jul 2001 (http://www.openssl.org/)
2003-01-21 21:11:17: INFO: isakmp.c:1365:isakmp_open(): 127.0.0.1[500] used
as isakmp port (fd=7)
2003-01-21 21:11:17: INFO: isakmp.c:1365:isakmp_open(): 10.0.1.1[500] used as
isakmp port (fd=8)
2003-01-21 21:11:17: INFO: isakmp.c:1365:isakmp_open(): 192.168.1.5[500] used
as isakmp port (fd=9)

2003-01-21 21:11:37: INFO: isakmp.c:1689:isakmp_post_acquire(): IPsec-SA
request for 192.168.2.5 queued due to no phase1 found.
2003-01-21 21:11:37: INFO: isakmp.c:794:isakmp_ph1begin_i(): initiate new
phase 1 negotiation: 192.168.1.5[500]<=>192.168.2.5[500]
2003-01-21 21:11:37: INFO: isakmp.c:799:isakmp_ph1begin_i(): begin Identity
Protection mode.
2003-01-21 21:11:37: INFO: vendorid.c:128:check_vendorid(): received Vendor
ID: KAME/racoon
2003-01-21 21:11:37: INFO: vendorid.c:128:check_vendorid(): received Vendor
ID: KAME/racoon
2003-01-21 21:11:38: INFO: isakmp.c:2417:log_ph1established(): ISAKMP-SA
established 192.168.1.5[500]-192.168.2.5[500] spi:6a01ea039be7bac2:bd288ff60eed
2003-01-21 21:11:39: INFO: isakmp.c:938:isakmp_ph2begin_i(): initiate new
phase 2 negotiation: 192.168.1.5[0]<=>192.168.2.5[0]
2003-01-21 21:11:39: INFO: pfkey.c:1106:pk_recvupdate(): IPsec-SA established:
ESP/Tunnel 192.168.2.5->192.168.1.5 spi=68291959(0x4120d77)
2003-01-21 21:11:39: INFO: pfkey.c:1321:pk_recvadd(): IPsec-SA established:
ESP/Tunnel 192.168.1.5->192.168.2.5 spi=223693870(0xd554c2e)
```

4 Fazit

Diese kleine Demonstration zeigt, dass der Aufbau virtueller privater Netzwerke mit dem kommenden Linux Kernel 2.6 ohne einen zusätzlichen FreeS/WAN Kernel Patch

mit einer alternativen Implementierung möglich sein wird. Diese Implementierung ist bereits voll funktionsfähig und wird in den nächsten Monaten bis zur Freigabe des Kernels zusätzliche Reife erfahren.

Racoon hat seine Interoperabilität als IKE Daemon bereits im Rahmen des KAME Projektes mehrfach unter Beweis gestellt und stellt eine sehr ausgereifte Implementierung zur Verfügung.

5 Links

- KAME Projekt: http://www.kame.net
- Linux Advanced Routing and Traffic Control Howto: http://www.lartc.org
- `setkey` und `racoon` Source:
  ftp://ftp.inr.ac.ru/ip-routing/iputils-ss021109-try.tar.bz2
- OpenBSD `isakmpd` Patch: http://www.thinknerd.de/~thomas/IPsec/
- RPM Pakete für setkey und racoon:
  http://www.spenneberg.org/VPN/Kernel-2_5-IPsec
- Letzte Version dieses Artikels: http://www.spenneberg.com/

6 Dank für Anregungen

- Stefan Gybas: Besseres Fontencoding