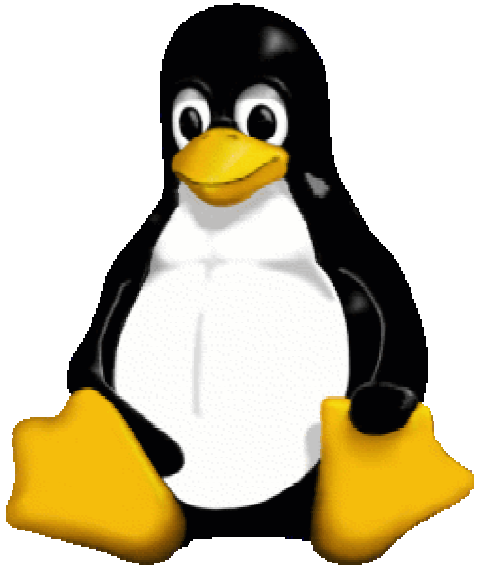
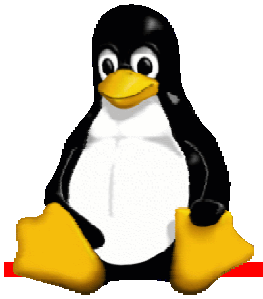


Sicurezza a livello IP: IPsec e Linux FreeS/WAN

Davide Cerri

Pluto Meeting 2001



Problematiche di sicurezza

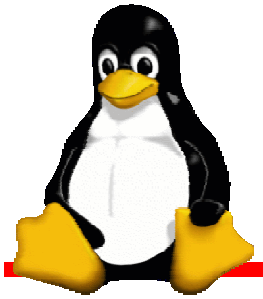
- Sicurezza può voler dire **diverse cose**...
  - riservatezza
  - autenticazione
  - integrità
  - disponibilità
  - autorizzazione
  - non ripudio
- Quando TCP/IP fu progettato non si fece attenzione ai problemi di sicurezza.

Sicurezza a diversi livelli

- Si possono introdurre le funzionalità di sicurezza a **diversi livelli** dello stack di rete:
  - applicazione: ad esempio PGP
  - tra applicazione e trasporto (sessione): TLS/SSL
  - rete: IPsec
- IPsec è la soluzione **più generale**, ma non è la soluzione a tutti i problemi!
  - IPsec protegge l'informazione durante il transito in rete tra due macchine.

Che cos’è IPsec

- IPsec è **parte integrante di IPv6**, ma può essere usato anche con IPv4 come estensione.
- È uno **standard IETF**.
- Non è un unico protocollo ma un’architettura di sicurezza a livello IP, composta da **diversi elementi**.
- Può essere utilizzato sia **end−to−end** che **gateway−to−gateway**.



Protocolli IPsec

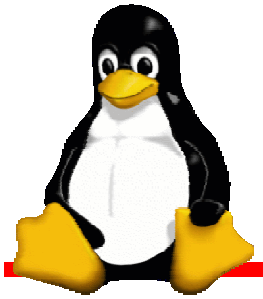
- **AH** (Authentication Header)
  autenticazione, integrità
- **ESP** (Encapsulating Security Payload)
  riservatezza, autenticazione, integrità
- **IKE** (Internet Key Exchange)
  scambio delle chiavi

Security Association

- Per utilizzare AH e/o ESP i due interlocutori devono aver prima negoziato una “**security association**” (SA).
- La SA è un “contratto” che specifica gli algoritmi crittografici e le relative chiavi, e qualsiasi altro parametro necessario alla comunicazione sicura.
- La negoziazione delle SA è compito del protocollo IKE.

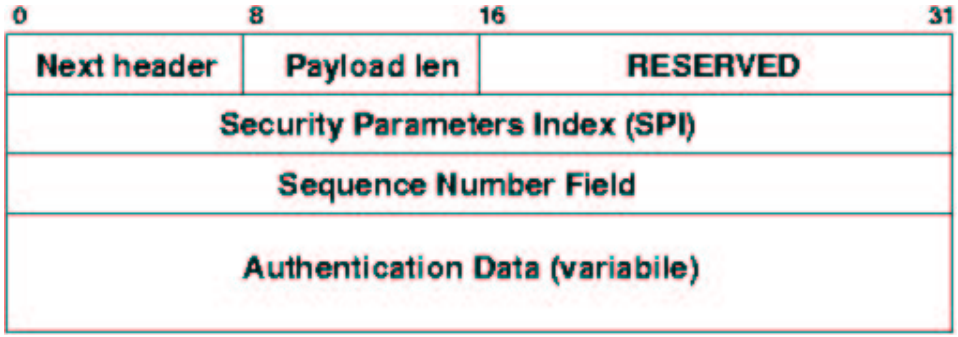
SPD e SAD

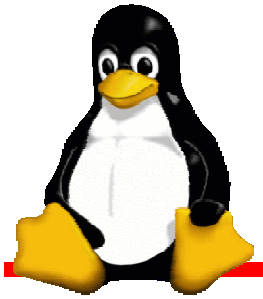
- L’architettura di IPsec comprende anche due basi di dati:
  - **SPD** (Security Policy Database)
    - contiene le policy IPsec, ovvero specifica, tramite una serie di selettori, quali trasformazioni vadano applicate a quale traffico.
  - **SAD** (Security Association Database)
    - contiene i dati relativi alle security association attive (chiavi, parametri, altri dati...).



Il protocollo AH

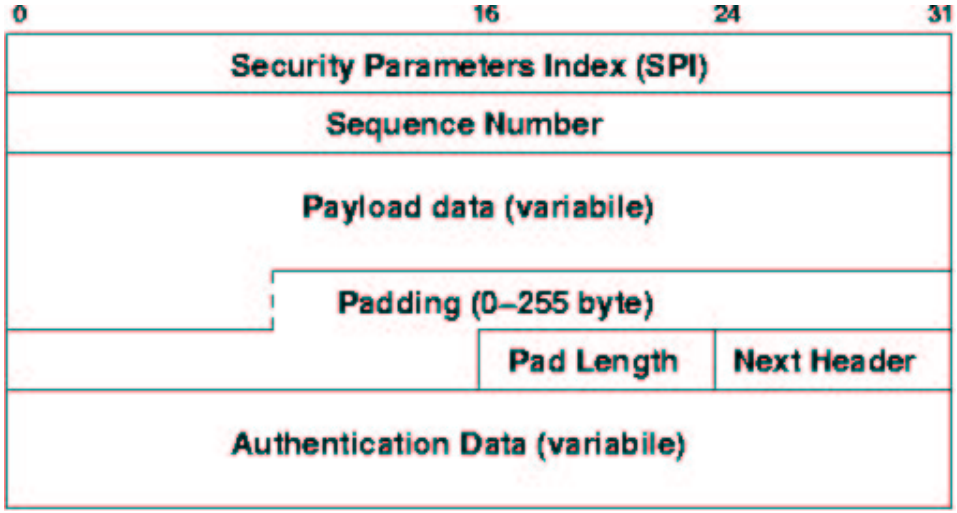
- **AH** (Authentication Header) fornisce servizi di **autenticazione, integrità e anti−replay**.
- L'autenticazione copre praticamente l'**intero pacchetto IP**.
  - sono esclusi solo i campi variabili dell'header IP (TTL, checksum...)

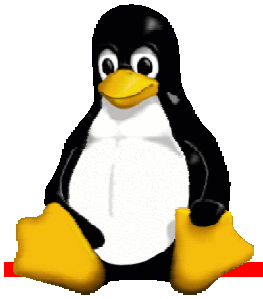




ESP

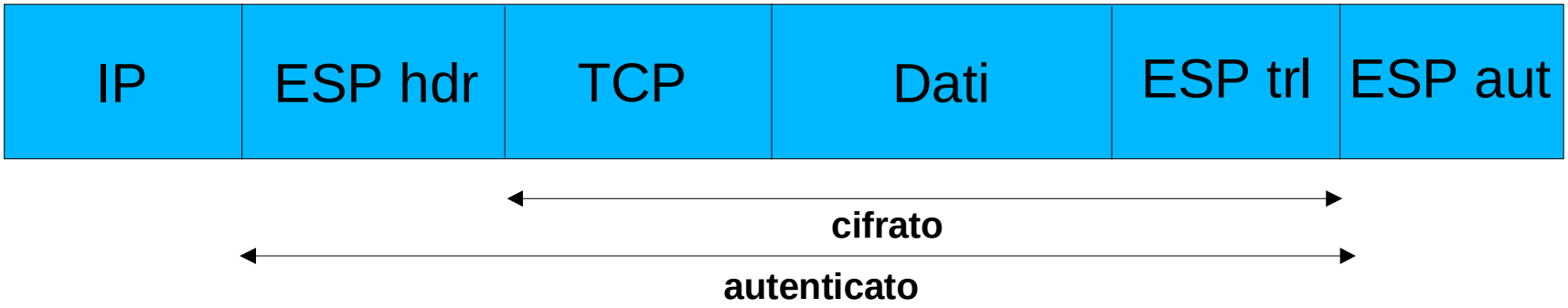
- **ESP** (Encapsulating Security Payload) fornisce servizi di **riservatezza, integrità, autenticazione e anti−replay**.
- ESP agisce su ciò che incapsula, quindi non sull'header IP esterno.

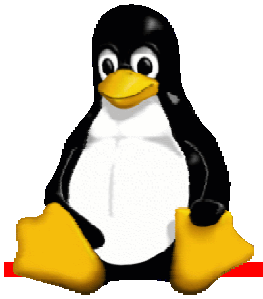




Modalità trasporto

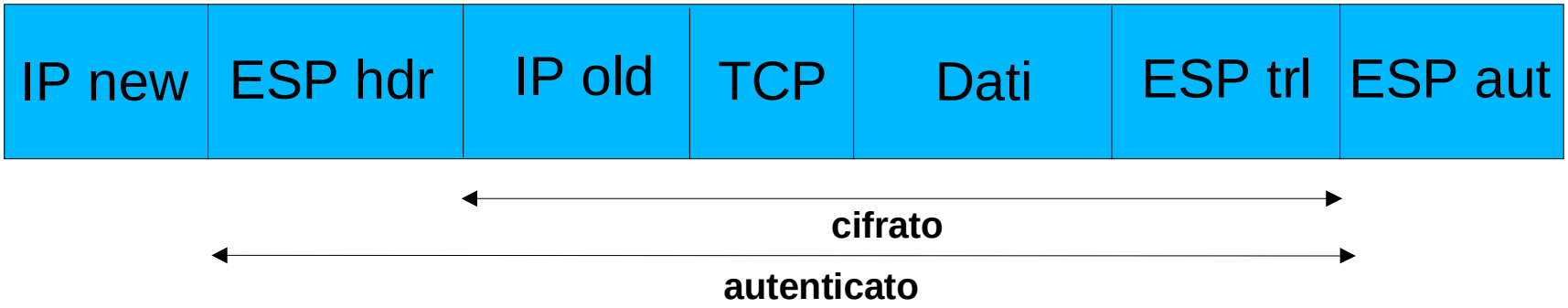
- Nella modalità **trasporto** (possibile solo tra host) gli header di AH e/o ESP sono inseriti tra l'header IP e quello di trasporto.

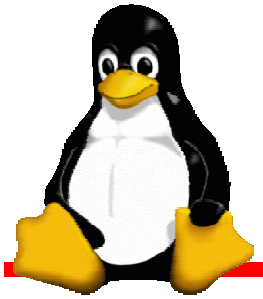




Modalità tunnel

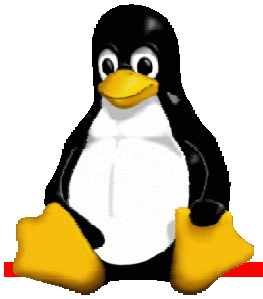
- Nella modalità **tunnel** il pacchetto IP originale viene incapsulato in un nuovo pacchetto IP





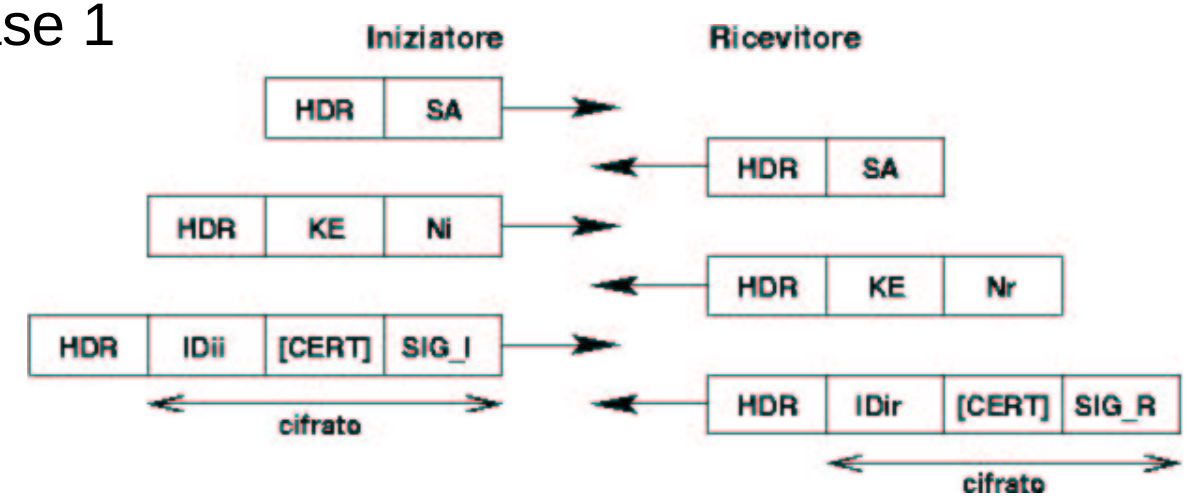
IKE

- **IKE** (Internet Key Exchange) è un protocollo di livello applicazione, che negozia le security association per AH/ESP.
- L'handshake avviene in **due fasi**:
  - prima fase: si crea una SA per IKE stesso (detta ISAKMP SA o IKE SA);
  - seconda fase: sfruttando la ISAKMP SA si creano le SA IPsec.

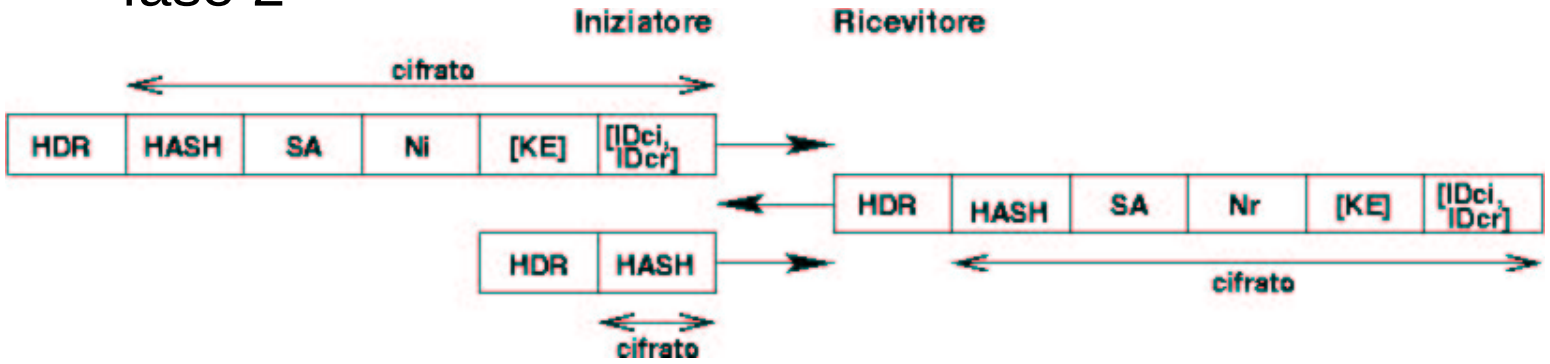


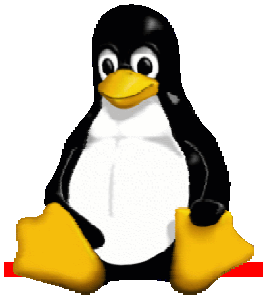
IKE: handshake completo

fase 1

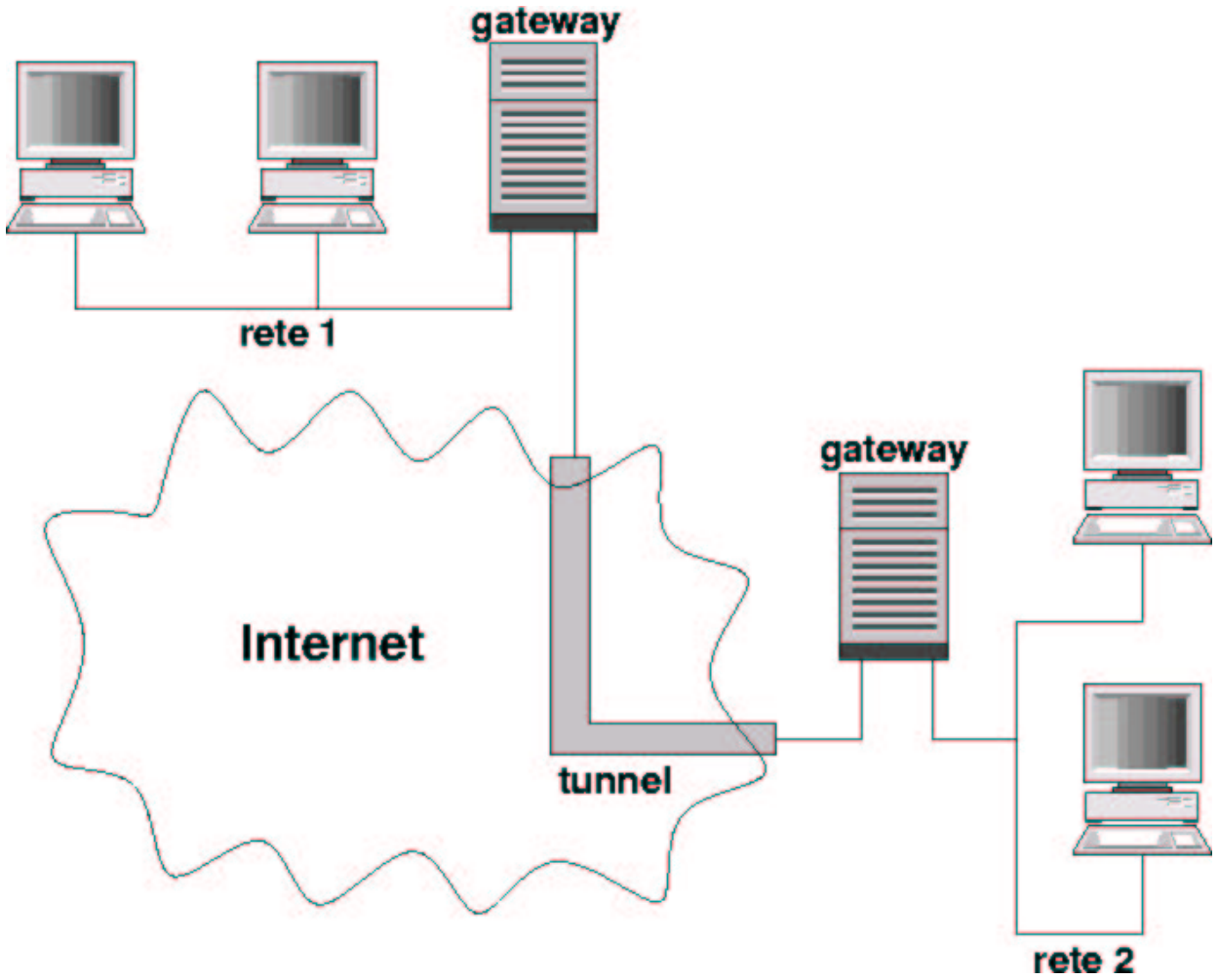


fase 2



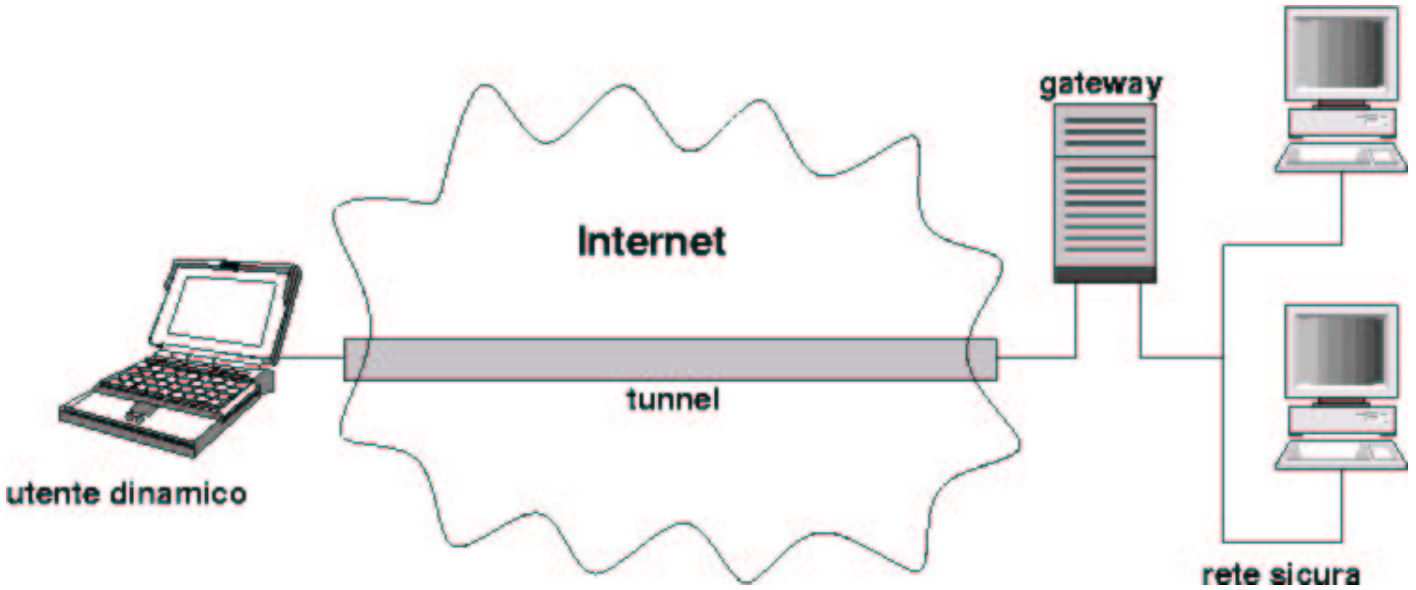


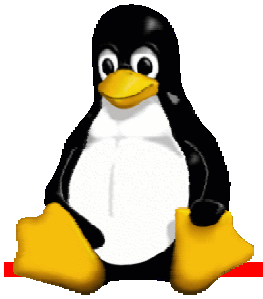
Rete privata virtuale



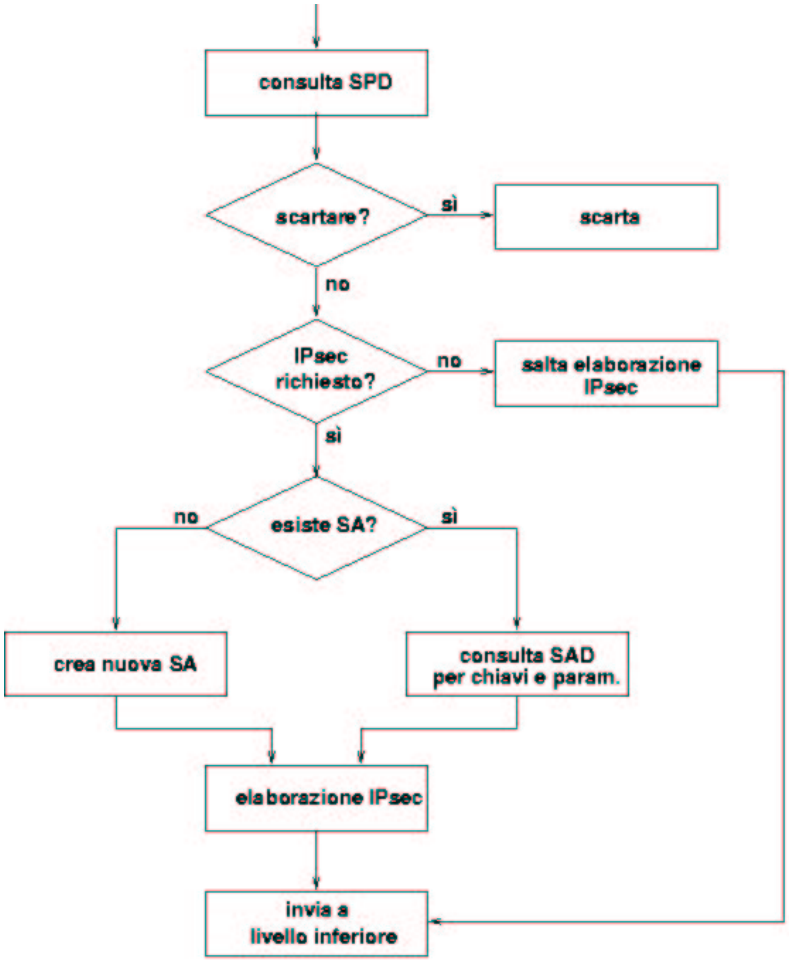
“Road warrior”

- Il “road warrior” si può vedere come un caso degenere di VPN, in cui da una parte c’è un unico host (ma: dinamico!).

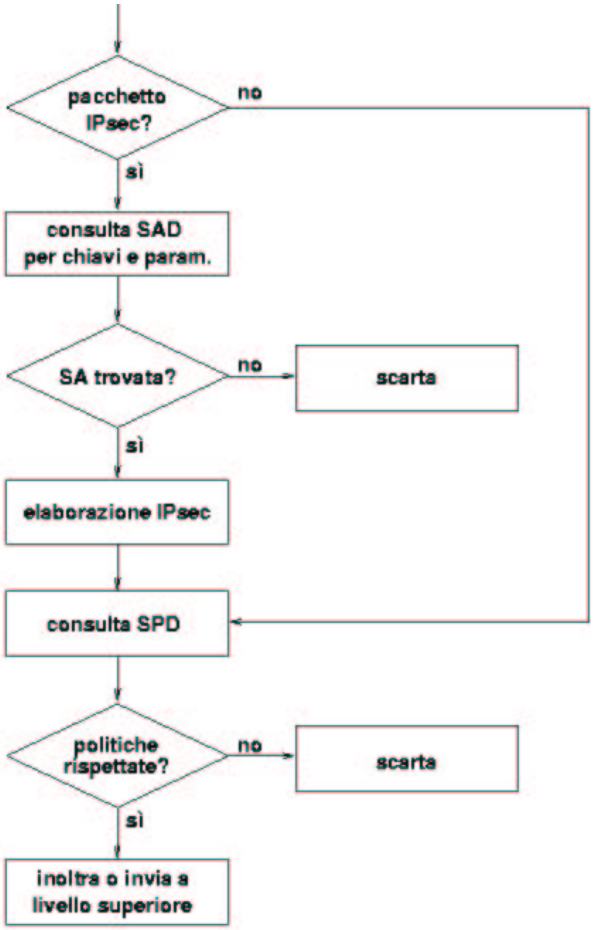




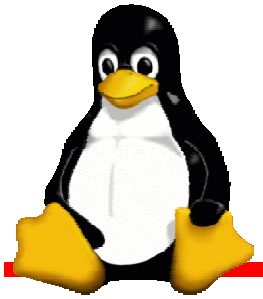
Elaborazione IPsec (in teoria)



(a) traffico in uscita (outbound)

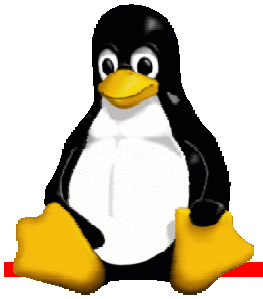


(b) traffico in ingresso (inbound)



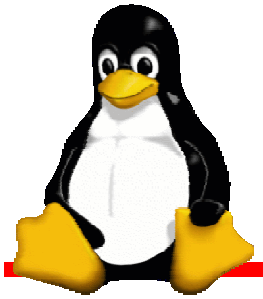
Implementazione di IPsec

- L'RFC dell'architettura di IPsec (2401) indica tre tipi di implementazioni di IPsec:
  - **nativa**: IPsec è integrato nell'implementazione nativa di IP;
  - "**bump in the stack**": IPsec è posto tra IP e le interfacce di rete;
  - "**bump in the wire**": IPsec è implementato in un dispositivo hardware esterno.



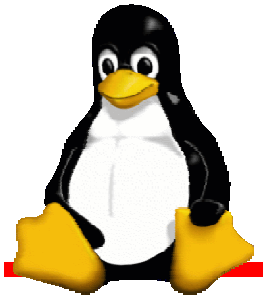
Linux FreeS/WAN

- FreeS/WAN è un'**implementazione di IPsec per Linux**.
- È composta da due parti:
  - **KLIPS**: parte kernel, implementa AH, ESP e la gestione dei pacchetti IPsec;
  - **Pluto**: demone in ascolto sulla porta UDP 500, implementa IKE.
- Per l'installazione è necessario ricompilare il kernel.



Linux FreeS/WAN

- L'implementazione è di tipo "**bump in the stack**":
  - si aggiungono delle **interfacce di rete virtuali** ipsec0, ipsec1...
  - si **modificano le tabelle di routing** in modo da instradare il traffico destinato all'elaborazione IPsec a queste interfacce.
- Le connessioni IPsec vengono normalmente attivate/disattivate tramite linea di comando oppure all'avvio di Pluto.



FreeS/WAN: caratteristiche

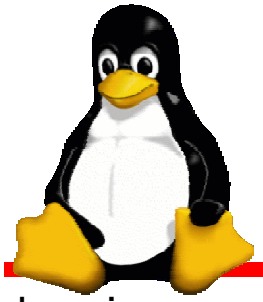
- FreeS/WAN supporta:
  - AH ed ESP (non ESP senza cifratura)
  - IKE main mode e quick mode (non aggressive mode)
  - Autenticazione mediante segreto condiviso o chiavi RSA (mediante certificati X.509 usando una patch)
  - 3DES per la cifratura
  - HMAC con MD5 o SHA1 per l’autenticazione
  - IPComp per la compressione del payload IP

“Opportunistic encryption”

- La “**opportunistic encryption**” è una caratteristica sperimentale di FreeS/WAN.
- Permette a due macchine che **non abbiano alcuna precedente informazione** l’una dell’altra di instaurare una connessione IPsec, andando a prendere la chiave pubblica dell’interlocutore da un **server DNS** (utilizzando, in futuro, DNSSEC − DNS Security Extensions).

“Opportunistic encryption”

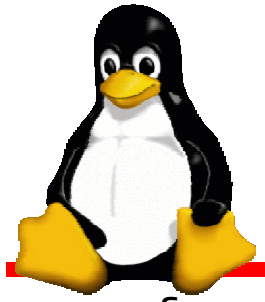
- Per inserire le chiavi pubbliche nel DNS si usa il record **KEY**.
- È previsto un particolare record **TXT** per indicare il proprio security gateway.
- Se la opportunistic encryption è abilitata, i pacchetti in uscita vengono intercettati e “**intrappolati**”, cercando nel frattempo di instaurare una SA con la destinazione (o con il suo gateway).



FreeS/WAN: ipsec.conf (1)

```
# basic configuration
config setup
    # THIS SETTING MUST BE CORRECT or almost nothing will work;
    # %defaultroute is okay for most simple cases.
    interfaces=%defaultroute
    # Debug-logging controls
    klipsdebug=none
    plutodebug=none
    plutoload=%search
    plutostart=%search
    # Close down old connection when new one using same ID shows up.
    uniqueids=yes

# defaults for subsequent connection descriptions
conn %default
    # How persistent to be in (re)keying negotiations (0 means very)
    keyingtries=0
    # RSA authentication with keys from DNS.
    authby=rsasig
    leftrsasigkey=%dns
    rightrsasigkey=%dns
```



FreeS/WAN: ipsec.conf (2)

automatic keying

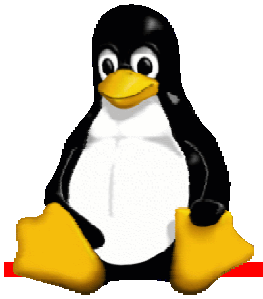
```
conn arfr
    left=192.168.20.20
    rightnexthop=192.168.30.3
    right=192.168.30.30
    leftrsasigkey=0sAQNjIuAgsnUm9BZNd6UY1GGhTbL[...]
    auto=add
```

manual keying

```
conn manual
    left=192.168.20.20
    rightnexthop=192.168.30.3
    right=192.168.30.30
    type=tunnel
    esp=3des-sha1-96
    spi=0x200
    espenckey=0x701022c3_fc92843f_2c5b9b0d_e5291031_6bf9326c_7235e2ff
    espauthkey=0x189383cb_970b21f3_900923bd_3a24cbe6_334faa78
    compress=no
```

opportunism

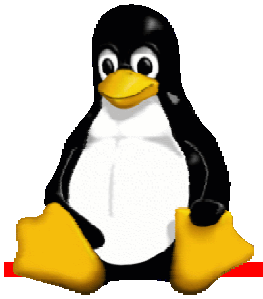
```
conn opp
    left=%defaultroute
    right=%opportunistic
    # uncomment to enable incoming; change to auto=route for outgoing
    auto=add
```



FreeS/WAN: connessione

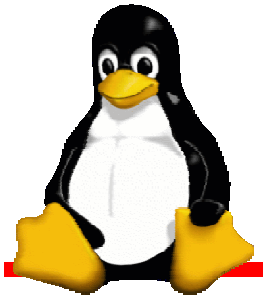
- Per avviare una connessione (con automatic keying) in generale:
  - `ipsec auto --add` *`nome`*
  - `ipsec auto --up` *`nome`*
    (opportunistic: `ipsec auto --route` *`nome`*)

```
[root@frodo /]# ipsec auto --up arfr
104 "arfr" #3: STATE_MAIN_I1: initiate
106 "arfr" #3: STATE_MAIN_I2: from STATE_MAIN_I1; sent MI2, expecting MR2
108 "arfr" #3: STATE_MAIN_I3: from STATE_MAIN_I2; sent MI3, expecting MR3
004 "arfr" #3: STATE_MAIN_I4: ISAKMP SA established
112 "arfr" #4: STATE_QUICK_I1: initiate
004 "arfr" #4: STATE_QUICK_I2: sent QI2, IPsec SA established
```



FreeS/WAN: stato

```
[root@frodo /]# ipsec look
frodo.etechlab.cefriel.it Tue Dec  4 17:08:18 CET 2001
192.168.30.30/32   -> 192.168.20.20/32   => tun0x1004@192.168.20.20
esp0xd5dc423@192.168.20.20  (0)
ipsec0->eth0 mtu=16260(1500)->1500
esp0xd2af355f@192.168.30.30 ESP_3DES_HMAC_MD5: dir=in  src=192.168.20.20
iv_bits=64bits iv=0xe77dc9cadbc87f5c ooowin=64 alen=128 aklen=128 eklen=192
life(c,s,h)=add(22,0,0)
esp0xd5dc423@192.168.20.20 ESP_3DES_HMAC_MD5: dir=out src=192.168.30.30
iv_bits=64bits iv=0xa237acb02e7cbfbd ooowin=64 alen=128 aklen=128 eklen=192
life(c,s,h)=add(22,0,0)
tun0x1003@192.168.30.30 IPIP: dir=in  src=192.168.20.20 life(c,s,h)=add(22,0,0)
tun0x1004@192.168.20.20 IPIP: dir=out src=192.168.30.30 life(c,s,h)=add(22,0,0)
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         192.168.30.3    0.0.0.0         UG       40 0           0 eth0
192.168.20.20   192.168.30.3    255.255.255.255 UGH      40 0           0 ipsec0
192.168.30.0    0.0.0.0         255.255.255.0   U        40 0           0 eth0
192.168.30.0    0.0.0.0         255.255.255.0   U        40 0           0 ipsec0
```



FreeS/WAN: sito ufficiale

- Per saperne di più su FreeS/WAN: http://www.freeswan.org

- Versione attuale: 1.93 (4 dicembre 2001)