IPSEC FOR LARGE SCALE DEPLOYMENTS

DevConf, Brno, January 2020

Presented by Paul Wouters,
VPN Technologies
THE LIBRESWAN PROJECT

An Internet Key Exchange (‘‘IKE’’) daemon for IPsec

- Continuation of openswan, which itself was a fork of freeswan
- Uses the NSS library for all its crypto
- Certified (FIPS, Common Criteria, USGv6, etc.)
- Supports X.509 and DNSSEC as PKI
- Supports raw public keys and PSK
- Supports IKEv1 and IKEv2
IPsec PRIMER
IKE + IPsec = VPN

IKE (USERLAND)
ISAKMP, IKE SA, PHASE 1
UDP PORT 500 AND 4500

• Command Channel
• Peer authentication
• Connection parameter negotiation
• IPsec symmetric key generation
• Communicates to kernel

• IKE itself is encrypted
• IKE does not encrypt the IP traffic

IPsec (KERNEL)
IPsec SA, CHILD SA, PHASE 2
PROTOCOL 50 (ESP) AND 51 (AH)

• Data Channel
  • Encapsulated Security Payload (ESP) IP packet encryption
  • Authenticated Header (AH)
  • ESPinUDP (for NAT)

• Tunnel Mode (IP in IP)
• Transport Mode
1. IPsec in the kernel has policies (SPD) and states (SAD)
   • Visible via: ip xfrm policy, ip xfrm state
   • Communication between userland and kernel visible via: ip xfrm monitor
   • Packets matching policies with a linked state causes encryption/decryption
   • Unencrypted packets matching an encryption policy without state are dropped
   • First outgoing plaintext packet matching IPsec policy without IPsec state triggers an ACQUIRE
2. Userland inserts “trap” policies for on-demand tunnels
3. Userland requests to receive ACQUIREs and processes these when received
   • Perform IKE negotiation with remote peer
   • Send IPsec policy and encryption/authentication keys to the kernel
4. Kernel processes netlink messages
   • Install received crypto keys in state, link crypto state to policy
   • If TCP was stored, send out cached TCP packet
LINUX IPsec IMPLEMENTATION (XFRM)

Mobile IKE: Support for mobile clients that change IP address

1. Libreswan receives an IKE message from a new IP address
2. Matches the encrypted IKE packet via SPI’s to an existing peer
3. Decrypts and confirms this IKE packet is a new packet (via sequence number)
4. Send XFRM_MIGRATE message to kernel
5. Kernel updates endpoint IP of the XFRM state
6. Libreswan confirms via a reply IKE message using the new destination IP address
src 10.3.230.191/32 dst 0.0.0.0/0
  dir out priority 666 ptype main
  tmpl src 76.10.157.68 dst 209.132.183.55
    proto esp reqid 16413 mode tunnel
src 0.0.0.0/0 dst 10.3.230.191/32
  dir fwd priority 666 ptype main
  tmpl src 209.132.183.55 dst 76.10.157.68
    proto esp reqid 16413 mode tunnel
src 0.0.0.0/0 dst 10.3.230.191/32
  dir in priority 666 ptype main
  tmpl src 209.132.183.55 dst 76.10.157.68
    proto esp reqid 16413 mode tunnel
src 209.132.183.55 dst 76.10.157.68
  proto esp spi 0x605ad2be reqid 16413 mode tunnel
  auth-trunc hmac(sha1) 0x4b7e46cdee9c27588a1a75f6846073cea 96
  enc cbc(aes) 0x11ddc9080945111087e81f9ebda5aacb7612c78af1895
src 76.10.157.68 dst 209.132.183.55
  proto esp spi 0x8ca00de3 reqid 16413 mode tunnel
  auth-trunc hmac(sha1) 0x1119585d334a88e023134a100eca6b09f 96
  enc cbc(aes) 0x310b852b9cbaf2cace7979c1aeb5df4b32eb418c5c300
2019: Libreswan

- RFC 8229 “TCP Encapsulation of IKE and IPsec packets” support [*]
- Proof of concept for per-cpu IPsec SA support (pCPU XFRM)
- XFRMi virtual interface support to replace VTI virtual interfaces
- libvirt/KVM based testing complimented with namespace based testing
  - Much faster, takes less resources, can run tests in parallel
- NSS: Use the new “IPsec profile” X.509 validation (RFC 4945)
- Dramatically improve IKE performance by offloading more code to threads
- NSS: Cache decrypted private keys (Bob’s performance trick)
- RFC 5685 IKEv2 REDIRECT support for dynamic load balancing
- CVE-2019-12312 (crasher) and CVE-2019-10155 (sig validation failure)
- Improvements for Enterprise-wise (mesh) encryption
XFRMi virtual interface

- Specify: ipsec-device=<num> will create device ipsecN
- tcpdump on ethX shows encrypted packets, on ipsecN shows plaintext
- Can be managed by the system (e.g., NetworkManager) or by libreswan
- Allow for routing based VPNs – if packet enters device, encrypt it
- Uses XFRM IF_ID (new IPsec policy parameter) and does not require MARKing
- Can be placed in a network namespace to enforce VPN-only access
- Some support in NetworkManager and systemd

- Older VTI (ip_vti0) implementation had limitations:
  - Each IPsec SA needed its own VTI interface and could only be ipv4 or ipv6
  - There could only be one wildcard interface (max 1 dynamic IP remote client)
  - VTI Only supported Tunnel Mode, not Transport Mode
  - Used GRE keys and MARKs
TCP support for Remote Access VPN

- IPsec (ESP) is usually encapsulated in UDP to traverse NATs
- IKE protocol uses UDP
- Some networks block IPsec (or all UDP except DNS)

- RFC 8229: TCP encapsulation of IKE and ESP
  - Port number not defined (evasive actions)
  - “IKETCP” prefix to support demultiplexing (aka IPsec-in-TLS support)

- New per connection options:
  - tcp-listen=<port>
  - tcp-remoteport=<port>
  - tcponly=yes|no
Per-CPU support for high speed IPsec SAs

- Proof of Concept for per-cpu IPsec SA support (pCPU XFRM)
- Currently, 1 IPsec SA can only use 1 CPU (about 1-5 gbps)
- With pCPU, create $\langle num\rangle$ CPU times identical IPsec SA’s
  - One IPsec SA is the main (catch all) IPsec SA
  - Further identical IPsec SA’s are installed for a specific CPU
- Processes generating traffic on one CPU, use the one IPsec SA
- Reduces out-of-order packets if multiple processes are sending data
- Lab result show CPU combined speed (eg 3 CPU * 1gbps/cpu results in 3gbps)
- TODO: per-cpu “fake” ACQUIRE messages for per-CPU on-demand support
- TODO: support for on-demand QoS IPsec SA’s
- See libreswan.org/wiki/XFRM_pCPU for kernel and libreswan code and docs
  - clones=$\langle num\rangle$
Libreswan testing

- KVM/libvirt testing - 700+ tests
- libreswan.org/wiki/Test_Suite_-_KVM
- Took between 4 and 8 hours to run - reduced to 2-3 hours
- Would regularly hang KVM instances
- Tests cannot run in parallel

- Introduced namespaces based testing
- libreswan.org/wiki/Test_Suite_-_Namespace
- Re-uses KVM/libvirt tests, so most of the 700+ tests are functional
- Tests can be started in parallel (but sadly not too many still)
IPsec deployment types

- Host to host VPN
- Site to site VPN
- Remote Access VPN
- Enterprise wide encryption (mesh encryption)
- Internet wide Opportunistic Encryption (try IPsec, fallback to clear)
Using self-signed or CA based certificates:

```plaintext
# install local certificate: ipsec import MyName.example.com.p12
# Convention used: left means local, right means remote
# /etc/ipsec.d/host-to-host.conf
# bring up manually using: ipsec auto –up host-to-host

conn host-to-host
  left=192.1.2.23
  leftid=@MyName.example.com
# our certificate
  leftcert=MyName.example.com
right=192.1.2.45
  rightid=@Peer.example.com  # ID must be SubjectAltName on cert
# their certificate transmitted via IKE
  auto=ondemand
```
Net to Net example configuration
Using self-signed or CA based certificates:

```bash
# install local certificate: ipsec import MyName.example.com.p12
# Convention used: left means local, right means remote
# /etc/ipsec.d/net-to-net.conf
# bring up manually using: ipsec auto –up net-to-net

conn net-to-net
left=192.1.2.23
leftsubnet=192.0.1.0/24
leftid=@MyName.example.com
# our certificate
leftcert=MyName.example.com
right=192.1.2.45
rightsubnets={192.0.2.0/24, 192.168.0.0/16}
rightid=@Peer.example.com  # ID must be SubjectAltName on cert
# their certificate transmitted via IKE
auto=ondemand
```
Remote Access Server configuration
Using CA based certificates:

```bash
# /etc/ipsec.d/vpnserver.conf

conn vpnserver
    left=193.110.157.148
    leftcert=vpn.nohats.ca
    leftid=@vpn.nohats.ca
    # for split-VPN, specify your server’s subnet instead of 0/0
    leftsubnet=0.0.0.0/0
    rightaddresspool=100.64.13.2-100.64.13.254
    right=%any
    rightid=%fromcert

# server options
    auto=add
    rekey=no
    modecfgdns=8.8.8.8
    modecfgpull=yes
    dpddelay=9m
    dpdtimeout=30m
    dpdaction=clear
    mobike=yes
    ipsec-interface=yes
```
Remote Access Client configuration

Using CA based certificates:

```
# Import my PKCS#12 cert: ipsec import letoams.nohats.ca.p12
# /etc/ipsec.d/corporate.conf

conn corporate-vpn
    left=%defaultroute
    leftcert=letoams.nohats.ca
    leftsubnet=0.0.0.0/0
    leftmodecfgclient=yes
    right=vpn.nohats.ca
    rightsubnet=0.0.0.0/0
    rightid=@vpn.nohats.ca
    narrowing=yes
    rekey=yes
    mobike=yes
```
Remote Access Client configuration
Using CA based certificates:

```
$ sudo ipsec auto --up vpn.nohats.ca
181 "vpn.nohats.ca"[1] 193.110.157.148 #1: initiating IKEv2 IKE SA
181 "vpn.nohats.ca"[1] 193.110.157.148 #1: STATE_PARENT_I1: sent v2I1, expected v2R1
182 "vpn.nohats.ca"[1] 193.110.157.148 #2: STATE_PARENT_I2: sent v2I2, expected v2R2 {auth=IKEv2
cipher=AES_GCM_16_256 integ=n/a prf=HMAC_SHA2_512 group=MODP2048}
002 "vpn.nohats.ca"[1] 193.110.157.148 #2: certificate verified OK
    E=info@nohats.ca,CN=vpn.nohats.ca,OU=Clients,O=No Hats
    Corporation,L=Ottawa,ST=Ontario,C=CA
002 "vpn.nohats.ca"[1] 193.110.157.148 #2: IKEv2 mode peer ID is ID_FQDN: '@vpn.nohats.ca'
003 "vpn.nohats.ca"[1] 193.110.157.148 #2: Authenticated using RSA with IKEv2_AUTH_HASH_SHA2_512
002 "vpn.nohats.ca"[1] 193.110.157.148 #2: received INTERNAL_IP4_ADDRESS 100.64.13.3
002 "vpn.nohats.ca"[1] 193.110.157.148 #2: received INTERNAL_IP4_DNS 193.110.157.148
005 "vpn.nohats.ca"[1] 193.110.157.148 #2: Received INTERNAL_DNS_DOMAIN: nohats.ca
002 "vpn.nohats.ca"[1] 193.110.157.148 #2: up-client output: updating resolvconf
002 "vpn.nohats.ca"[1] 193.110.157.148 #2: negotiated connection [100.64.13.3-100.64.13.3:0-65535 0] ->
    [0.0.0.0-255.255.255.255:0-65535 0]
004 "vpn.nohats.ca"[1] 193.110.157.148 #2: STATE_V2_IPSEC_I: IPsec SA established tunnel mode
    {ESP/NAT=>0x0bb71fd2 <0x899a9f05 xfrm=AES_GCM_16_256-NONE NATOA=none NATD=193.110.157.148:4500
    DPD=passive}
```
Remote Access Client NetworkManager plugin

networkmanager-libreswan
## Libreswan Administration

### Select 4. Generate OR Revoke User Certificate to change

<table>
<thead>
<tr>
<th>Action</th>
<th>USERNAME</th>
<th>EMAIL VERIFIED</th>
<th>CERTIFICATE REVOKED</th>
<th>CERT NAME</th>
<th>CERT PASSWORD</th>
<th>KEY NAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generate user certificate</td>
<td>adnco</td>
<td>✔️</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>abrissaf</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>trialuser</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>abcsuser</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>newuser</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>rishabh</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>✔️</td>
</tr>
</tbody>
</table>

6.4. Generate OR Revoke User Certificates

Preview: github.com/Rishabh04-02/Libreswan-managing-interface
Remote Access Server interface

- Setup a Certificate Agency CA
- Setup the VPN server (TLS and IPsec)
- Generate client certificates (PKCS#12) for download
- Generate iOS/OSX VPN profiles (1 click install)
- Traffic Accounting of clients
- Temporarily and permanently Revoke client certificates
Enterprise wide (mesh) encryption

- Each node attempts on-demand (opportunistic) IPsec to each other node
- A node policy can be REQUIRE, OPTIONAL, or NO encryption using IPsec
- Nodes should be able to list policies based on IP ranges and/or ports
- Exceptions should be allowed for specific nodes
- Added a new node should not require any reconfiguration on existing nodes
- Idle IPsec connections should be removed over time
- Currently support X.509 and DNSSEC as PKI for authenticating nodes

- WARNING: Your network based firewalls and IDS are bypassed!
Enterprise wide (mesh) encryption

Group policy files in /etc/ipsec.d/policies/* list network CIDRs and ports

- /etc/ipsec.d/policies/clear - Only allow cleartext
- /etc/ipsec.d/policies/clear-or-private - Default clear, allow IPsec
- /etc/ipsec.d/policies/private - Require IPsec
- /etc/ipsec.d/policies/private-or-clear - IPsec, fallback to clear

# /etc/ipsec.d/policies/private-or-clear
193.110.157.0/24
193.111.228.0/24

# /etc/ipsec.d/policies/private
10.0.0.0/8
192.168.0.0/16

# /etc/ipsec.d/policies/clear
193.110.157.0/24:22 # ssh
193.110.157.0/24:443 # https
Enterprise wide (mesh) encryption
Policy files are implemented using “regular” connection definitions

```
# install localcertificate: ipsec import node1.example.com.p12
# /etc/ipsec.d/private.conf

conn private
  left=%defaultroute
  leftid=%fromcert
# our certificate
  leftcert=node1.example.com
  right=%opportunisticgroup
  rightid=%fromcert
  failureshunt=drop
  negotiationshunt=hold
  auto=ondemand
```
Enterprise wide (mesh) encryption

Configuration for IPsec with fallback to cleartext

```plaintext
# install localcertificate: ipsec import node1.example.com.p12
# /etc/ipsec.d/private-or-clear.conf

cconn private-or-clear
  left=%defaultroute
  leftid=%fromcert
  leftcert=node1.example.com
  right=%opportunisticgroup
  rightid=%fromcert
  failureshunt=passthrough
  negotiationshunt=passthrough
  # to not leak cleartext during IKE negotiation:
  # negotiationshunt=hold
  auto=ondemand
```

IPsec for large scale deployments
Enterprise wide (mesh) encryption

Configuration for preferring plaintext but accepting IPsec if requested

```bash
# install localcertificate: ipsec import node1.example.com.p12
# /etc/ipsec.d/clear-or-private.conf

conn clear-or-private
  left=%defaultroute
  leftid=%fromcert
  # our certificate
  leftcert=node1.example.com
  right=%opportunisticgroup
  rightid=%fromcert
  failureshunt=passthrough
  negotiationshunt=passthrough
  # to not leak cleartext during IKE negotiation:
  # negotiationshunt=hold
  # don’t initiate, but allow responding
  auto=add
```
Recap: Enterprise wide (mesh) encryption

• Create policy connections in /etc/ipsec.d/*.conf for IPsec policy groups
• Place IP address/port ranges in /etc/ipsec.d/policies/*
• Import host PKCS#12 certificate into libreswan
• Start the IPsec service
• Profit!

• Check encryption
  • ipsec trafficstatus
  • ipsec status
  • ipsec shuntstatus  (for remote peers we tried and failed IPsec to)
Internet wide mesh encryption using LetsEncrypt
(Requires libreswan 3.30, uses NAT-within-IPsec)

# setup on laptop
# (host initiated as anonymous, authenticates remote server)
$ sudo ipsec letsencrypt --client
# echo "193.110.157.131/32" >> /etc/ipsec.d/policies/private-or-clear
# echo "0.0.0.0/0" >> /etc/ipsec.d/policies/private-or-clear

# Setup on server:
# (host authenticates itself via FQDN (eg letsencrypt.nohats.ca)
$ sudo ipsec letsencrypt --server
# (create a letsencrypt certificate for IPsec)
$ sudo ipsec letsencrypt --generate-certificate my.example.com
Anonymous client to authenticated server

```bash
$ ping letsencrypt.libreswan.org
PING letsencrypt.libreswan.org (193.110.157.131) 56(84) bytes of data.
64 bytes from letsencrypt.libreswan.org (193.110.157.131): icmp_seq=2
ttl=64 time=96.2 ms
64 bytes from letsencrypt.libreswan.org (193.110.157.131): icmp_seq=3
ttl=64 time=96.7 ms

ipsec whack --trafficstatus
006 #2: "private-or-clear#193.110.157.131/32"[1]  
100.64.0.2/32=== ...193.110.157.131, type=ESP, add_time=1471926595,
inBytes=252, outBytes=252, id='CN=letsencrypt.libreswan.org',
lease=100.64.0.2/32
```
Eliminating the IP address conflicts caused by NAT

193.110.15.131 Remote Opportunistic IPsec server
192.168.2.45 Opportunistic Client pre-NAT IP address
100.64.0.2 IP address from IPsec server address pool

# ip xfrm pol
src 100.64.0.2/32 dst 193.110.157.131/32
dir out priority 2080 ptype main
tmpl src 192.1.2.45 dst 193.110.157.131
proto esp reqid 16389 mode tunnel
src 193.110.157.131/32 dst 100.64.0.2/32
dir fwd priority 2080 ptype main
tmpl src 193.110.157.131 dst 192.1.2.45
proto esp reqid 16389 mode tunnel
src 193.110.157.131/32 dst 100.64.0.2/32
dir in priority 2080 ptype main
tmpl src 193.110.157.131 dst 192.1.2.45
proto esp reqid 16389 mode tunnel
src 192.168.2.45/32 dst 193.110.157.131/32
dir out priority 2080 ptype main
tmpl src 192.1.2.45 dst 193.110.157.131
proto esp reqid 16389 mode tunnel
IPsec for large scale deployments

Eliminating the IP address conflicts caused by NAT

193.110.15.131  Remote Opportunistic IPsec server
192.168.2.45    Opportunistic Client pre-NAT IP address
100.64.0.1      Client IP address assigned by Opportunistic IPsec server

# iptables -t nat -L -n

Chain PREROUTING (policy ACCEPT)
  target     prot opt source               destination
  DNAT       all  --  193.110.157.131      100.64.0.1 \  
                  policy match dir in pol ipsec to:192.168.2.45

Chain POSTROUTING (policy ACCEPT)
  target     prot opt source               destination
  SNAT       all  --  0.0.0.0/0            193.110.157.131 \  
                  policy match dir out pol ipsec to:100.64.0.1

Basically: NAT within the IPsec subsystem
QUESTIONS & ANSWERS