Where the heck is Rapperswil?
HSR - Hochschule für Technik Rapperswil

- University of Applied Sciences with about 1500 students
- Faculty of Information Technology (300-400 students)
- Bachelor Course (3 years), Master Course (+1.5 years)
Agenda

• Overview of current strongSwan active/active HA solution

• Proposed XFRM Extensions
  • Enforcing policies for inbound transport mode SAs
  • Different timeouts for acquire states and SPIs
  • Query available algorithms via XFRM
  • ESP in UDP encapsulation for IPv6
  • Proper way to handle virtual IPv6 addresses
  • Marking inbound traffic after decryption
The strongSwan Project

IPsec Workshop Dresden, March 26-28 2018

Current Active/Active HA Solution
High Availability Design Goals

- Transparent to VPN clients
- No extensions to the IKEv2 protocol required
- No explicit synchronization of ESP sequence numbers between redundant gateways
- Both Active/Passive (Hot-Standby) and Active/Active (Load Sharing) scenarios to be supported
HA Solution using ClusterIP Mechanism

ClusterIP:
hash = jhash_2words(daddr.a4, spi)

Internet

Intranet

multicast clustermac

moon

Heartbeat
SA Updates

plaintext

vénus

Ciphertext (ESP)

multicast clustermac

mars (Virtual VPN Gateway)

Segment 1
Segment 2
...
Active/Passive Scenario with 1 ClusterIP Segment

Diagram showing two segments connected to an Intranet and Internet, with a 'moon' and 'venus' node in each segment.
Active/Active Scenario with 2 ClusterIP Segments

Segment 1

Segment 2

moon

venus

Segment 1

Segment 2

mars

Internet

Segment 1

Segment 2

Segment 1

Segment 2

mars

Internet
Two New Netfilter Hooks: XFMR_IN/XFRM_OUT

- PREROUTING
  - XFMR_OUT
    - Encrypt
  - INPUT
  - Decrypt

Plaintext → Encrypt → XFMR_OUT → XFMR_IN → Decrypt → Ciphertext (ESP)

Netfilter Flow
Changes to ClusterIP Module

- **Extended ClusterIP hash:** \( jhash\_2\text{words}(daddr.a4, \text{spi}) \)

- **Inbound packet handling**
  - SA lookup to determine SPI
  - **Responsible for segment:** Decrypt ESP packet and update anti-replay window
  - **Not responsible for segment:** Decrypt every 16\(^{th}\) ESP packet, update anti-replay window and drop packet

- **Outbound packet handling**
  - Policy/SA lookup to determine SPI and destination address
  - Increase sequence number
  - **Responsible for segment:** Encrypt packet
  - **Not responsible for segment:** Drop packet
Next Generation HA?

- IPv6 not supported by ClusterIP
- HA kernel patch against a moving Linux kernel target
- Possibility of a Linux kernel upstream solution?
- Switch from ClusterIP to xt_cluster which supports IPv4 and IPv6
- Other ideas?
The strongSwan Project

IPsec Workshop Dresden, March 26-28 2018

Proposed XFRM Extensions
Enforcing Policies for Inbound Transport Mode SAs

- Currently the Linux kernel does not enforce policies for IPsec transport mode.
- Policy: TCP *:80 -> Peer can send other protocols or to other ports
- Patch by Tobias posted 2014 on netdev mailing list.
Different Timeouts for Acquire States and SPIs

• Currently, SPIs allocated with XFRM_MSG_ALLOCSPi expire after the same timeout that is also used for the temporary states allocated after sending an acquire to the IKE daemon (/proc/sys/net/core/xfrm_acq_expires).

• However, keeping acquire states around that long might not be desired (e.g. in the trap-any scenario, although a populate-from-packet feature could help here too).

• Using the lifetime config on struct xfrm_userlsa_info that's part of struct xfrm_userspi_info this could easily be implemented.

• Patch by Tobias sent a year ago to Steffen Klassert.
Query Available Algorithms via XFRM

• To prepare an automatic ESP proposal it would be necessary to query the algorithms the kernel supports via XFRM. Similar to the feature provided by PF_KEY via `xfrm_probe_algs()`, however, that’s not actually that useful because it’s based on a static list.

• Ideally, we’d get a list of actually usable algorithms (modules? FIPS mode?)
UDP Encapsulation of ESP for IPv6

• UDP encapsulation of ESP is supported for IPv4 but strangely not for IPv6 even though natting IPv6 has been possible for a while.
• For us it is mainly of interest because our Android app requires UDP encapsulation to work in userland.
• With the upcoming TCP encapsulation this might be less of a problem, but it's usually preferable to use UDP encapsulation over TCP encapsulation.
• POC patch by Tobias available.
• Handling of UDP header checksum (RFC 6935/RFC 6936)?
Proper Way to Handle Virtual IPv6 Addresses

• We currently install virtual IPv6 addresses received from a server on a local interface and install specific source routes with that address and the remote subnets.

• The address is marked deprecated, the idea being that the kernel will only use this address for the explicit routes but not when doing address selection for other destinations.

• The question is whether this is the proper way of doing this.
Marking Inbound Traffic After Decryption

• Similar to the new outbound mark that's applied after encryption (XFRMA_OUTPUT_MARK) we'd like to discuss the possibility of adding a similar feature that applies a mark to inbound packets right after decryption.

• This would simplify applying a mark to specific tunnels (e.g. for QoS) without having to mark before encryption or based on possibly dynamic values like SPI/reqid.

• Patch by Steffen Klassert exists.