fixing/updating Documentation/networking/xfrm*

Linux IPsec Workshop
July 17, 2026
Vienna, Austria
Michael Richardson
<mcr+ietf@sandelman.ca>

Who am I?

Xelerance Corp 2003-2007,2014-2018

Internet technologist, doing IP since 1988. “Garage Entrepreneur”

SOLIDUM (1998-2001)

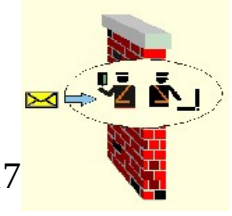
SANDELMAN SOFTWARE WORKS 1996-

FreeS/WAN (2001-2004)

SIMtone CORPORATION (2007-2009)

CREDIL CENTRE DE RECHERCHE ET DÉVELOPPEMENT EXPÉRIMENTAL EN INFORMATIQUE LIBRE / CENTRE FOR RESEARCH AND EXPERIMENTAL DEVELOPMENT IN INFORMATICS LIBRE 2009-2017

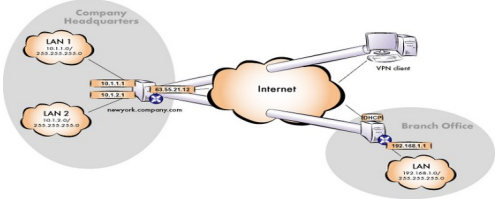
#4 at Milkyway Networks (1994)



ROLL – RFC6550 2012-

6TiSCH

RFC4322
RFC4025
RFC5386
RFC8415
RFC7416
RFC8366
RFC8995(BRSKI)
constrained-BRSKI



IETF standard security:IPsec/VPN

20+ RFCs

What's the Problem?

- the history is that person who implements Linux IPsec extension FOO, also writes some FOO support for IKE daemon XYZ.
  - and/or for iproute2
- other people then cargo cult/read code, try to figure out how stuff works
  - this was me on vti vs xfrmtun in 2024. If vti.rst had said: "deprecated do not use"
- Documentation/networking/xfrm* ~~is~~ was *almost* completely useless
  - new xfrm subdir from 2025, new file from Antony this year... progress
- implementing new extensions is harder than it needs to be, because it is hard to know which extensions are good, supported, vs bad ideas, poorly done, ...
- shit is getting more and more complex, and ESPv3 is gonna make it even worse...

MCR said he'd work on this in... 2023?
Then fell ill.

Proposal: explain how to use things

So in 2025, networking/xfrm* -> xfrm/stuff.

Antony did first commit of migrate documentation in May 2025

File namin quibble: _ vs - in file names... is there advice somewhere? _ seems like snake_case?

I SUGGEST:

- xfrm-overview.rst
- xfrm/
  - architecture.rst
  - caveat-ipcomp.rst
  - device-offload-overview.rst
  - extensions.rst
  - netlink-overview.rst
  - proc-stats.rst
  - sync-sa.rst
  - xfrm-roadmap.rst

Series of files:

- xfrm_msg_newsa.rst
  - (or xfrm-msg-newsa)
- xfrma_sa_extra_flags.rst
- links between them

Proposal: old files/rename

So do we do:

- Documentation/networking/xfrm*
- Documentation/networking/xfrm/stuff*

  File namin quibble: _ vs - in file names... is there advice somewhere? seems like snake_case?

Old files:

2. xfrm_device.rst: it seems really really stale?
   Suggest it becomes xfrm/device-offload-overview.rst

3. xfrm_proc.rst -> xfrm/proc-stats.rst
   Connection to private MIB/SNMP.
   Does anyone use this as part of a *MIB* (or YANG module),
   today?

4. xfrm_sync.rst -> xfrm/sync-sa.rst
   But, most of this belongs into messages.

Initial Hack at Documentation

- rebased this morning, pushed it:
  - https://code.credil.org/public-repos/linux.git/
  - branch https://code.credil.org/public-repos/linux.git/
  - based upon klassert ipsec-next-2026-06-12
- wrote up template **asciidoc**-ish in .rst for NEWSA, next three slides

```
// -*- mode: markdown; -*-

XFRM_MSG_NEWSA(1)
=================
:doctype: manpage

NAME
----
xfrm\_msg\_newsa - Adds a new Security Association (SA) to the SA database

APPLICABILITY
-------------
This message can create ESP (RFC4303), AH (RFC4302) security associations in
tunnel, transport or BEET (I-D.ietf-ipsecme-ikev2-beet-mode).

STRUCTURE
---------

    struct {
        struct nlmsghdr         n;
        struct xfrm\_userpolicy\_info  xpinfo;
        char                buf[RTA\_BUF\_SIZE];
    }
```

Probably unwelcome, “rst” -> Rust in emacs

```
    struct xfrm\_userpolicy\_info {
        struct xfrm\_selector    sel;
        struct xfrm\_lifetime\_cfg   lft;
        struct xfrm\_lifetime\_cur   curlft;
        \_\_u32             priority;
        \_\_u32             index;
        \_\_u8              dir;
        \_\_u8              action;
    #define XFRM\_POLICY\_ALLOW  0
    #define XFRM\_POLICY\_BLOCK  1
        \_\_u8              flags;
    #define XFRM\_POLICY\_LOCALOK1   /* Allow user to override global
policy */
        /* Automatically expand selector to include matching ICMP
payloads. */
    #define XFRM\_POLICY\_ICMP   2
        \_\_u8              share;
    };
```

backslash, because _italic_

```
DESCRIPTION
-----------

The MSG\_NEWSA and MSG\_UPDSA message are used to create or update a Security
Association.  The bulk of the historic contents of the SA xfrm\_userpolicy\
_info
structure, but a series of attributes (XFRMA) are required to provide
algorithm and key information, as well as newer attributes such as
Extended Sequence Number (ESN).

ATTRIBUTES REQUIRED
-------------------

(These are the ones that libreswan's netlink\_add\_sa() sends, as of 2026-07-
17):

XFRMA\_SA\_EXTRA\_FLAGS -  (FILE REFERENCE)

XFRMA\_REPLAY\_ESN\_VAL -
....
```

Other things in uapi headers

```
/* Structure to encapsulate addresses. I do not want to use
"standard" structure. My apologies. */
```

- MCR: be nice to know who "I" is/was, and whether we can just “fix” this union, and remove the comment.

Git annotate says:

- 1da177e4c3f41 (Linus Torvalds 2005-04-16 15:20:36 -0700 13) * "standard" structure. My apologies.

Other things in uapi headers
#define vs enum

```
    XFRM_MSG_NEWSA = 0x10,
#define XFRM_MSG_NEWSA XFRM_MSG_NEWSA

...

    XFRM_MSG_NEWSPDINFO,
#define XFRM_MSG_NEWSPDINFO XFRM_MSG_NEWSPDINFO
```

This is done, one assumes, because there are some userspace (IKE daemon, etc) that conditionally compile in support for some things based upon availability of the #define (#ifdef) in the `/usr/include` header.

This is a mistake

1. this seems harder for anything that is not C/C++ (Rust, Java, GO, Python/Ruby extensions)

2. host/OS where code is built... no relationship to where it is running

3. do we delete the #define, if/when the kernel drops support for that thing?

Runtime learning about messages and attributes

1. Create a new MSG which returns a list of messages and attributes (XFRMA_*)
2. The list of messages and attributes could be growing bit map of which MSG numbers and attributes are implemented/available in that kernel.
3. Or maybe it's a list of integers.

Probably this is a kernel compile time constant, based upon build options.

- Are there messages and attributes which can be added via loadable module? (out of tree?)
  - xfrm_user.c, xfrm_compat.c process XFRM_MSG_ and XFRMA_ via switch, array... not extensible **TODAY**. Do we think it will change?
- userspace can do this query, and then can adapt, usually enabling new features based upon availability.
- Note that distro maintainers often backport code from newer kernels to stable kernels, and sometimes these are necessary to solve security issues.