Christian Hopps
LabN Consulting, LLC

IP Traffic Flow Security

Improving IPsec Traffic Flow Confidentiality

IETF 107 – “draft-ipsecme-iptfs-01”

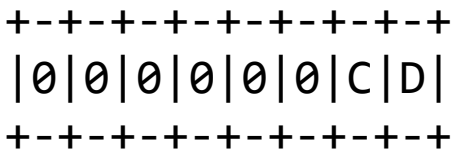
Update Since IETF 106

- draft-ietf-ipsecme-iptfs-01 published March 2, 2020
  - Prior to IETF 107 to address IETF 106 comments.
- Notable Changes
  - IKEv2 Transform changed to Notification
  - Added Sub-Type octet

IKEv2 USE_IPTFS Notification

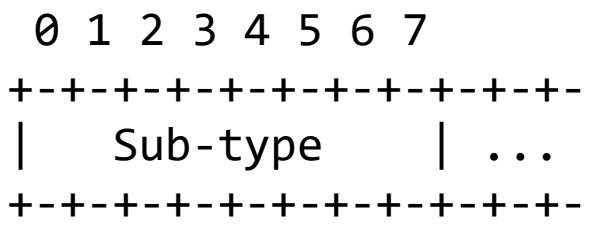
- Use notification during IKE_AUTH and CREATE_CHILD_SA for enabling IPTFS.
- Similar to USE_TRANSPORT_MODE (et al.) method.
- Required flags payload
- If required flags are not understood or supported then IPTFS mode is not enabled by responder or initiator deletes now established SA.

IKEv2 USE_IPTFS Notification Required Flags



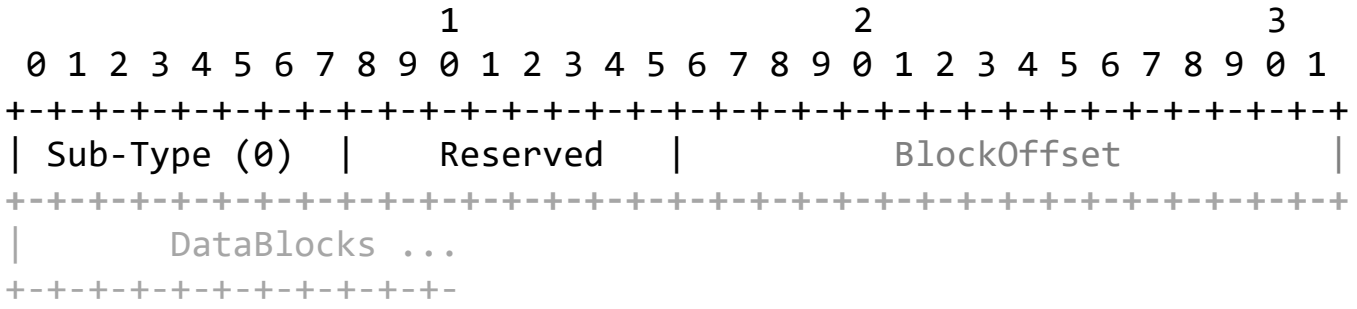
- **C** :: Congestion control bit. If set, the sender is requiring that congestion control information MUST be returned to it periodically
- **D** :: Don’t Fragment bit. if set, the sender of the notify message does not support receiving packet fragments

IPTFS_PROTOCOL Payload Format



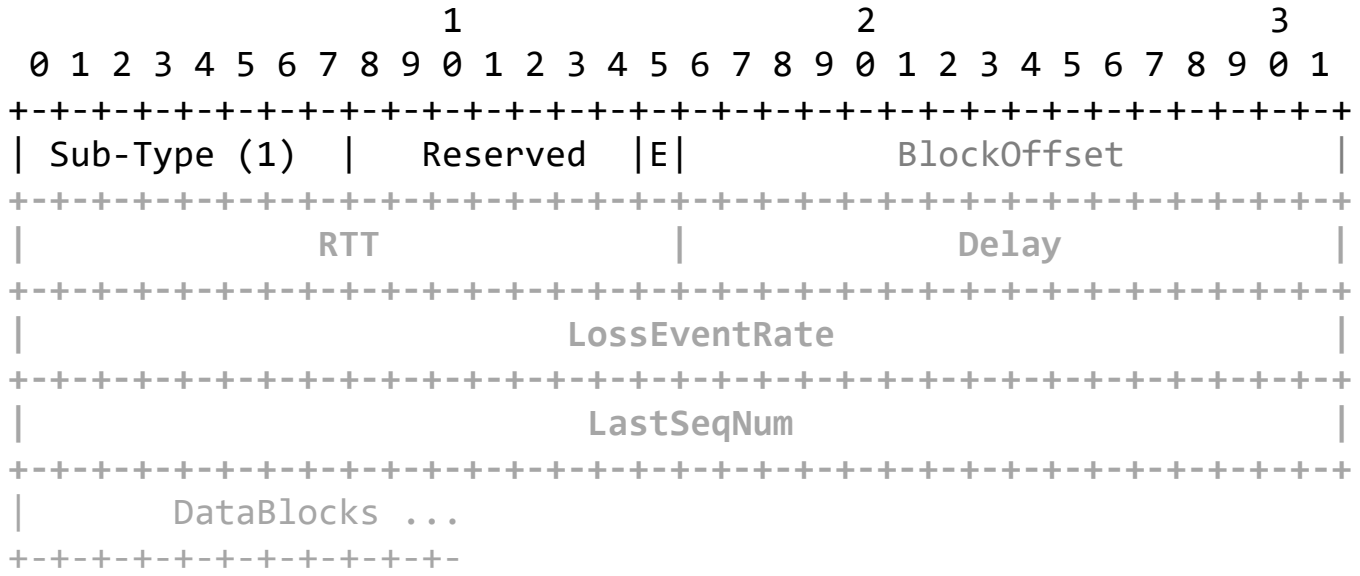
- **Sub-Type** :: An octet indicating the payload format.

Non-Congestion Control Payload Format



- **Sub-Type** :: An octet (value 0) indicating this payload format.

Congestion Control Payload Format



- **Sub-Type** :: An octet (value 1) indicating this payload format
- **E** :: ECN bit were used in calculating the **LossEventRate**
  - Same definition as before, just moved

Open Issues/Last Meeting Comments

- IP Number
  - Discussed on list a couple times. Waiting for chairs to forward the request.
  - Summary:
    - Use WESP consumes bandwidth, still have need for next-header number.
    - Get a number, start process early, our use is valid, IETF process should not block technically better choices.
    - Can fallback to overloading another IP protocol number for ESP only use.
- Transport Mode
  - To be defined in separate document
  - Will not conflict with this tunnel mode based document
    - sub-type, flags, or the mode itself can be used to differentiate any header changes

Other Issues/Notes

- Datablock (inner packet) alignment.
  - Con: Complicates encap/decap specification and code
  - Con: Wastes bandwidth
  - Pro: Aligning internal packets allows less rigorous whitebox code to work.
    - Ends up not being an issue as copy-out of packet header is required even when using indirect buffer chains.
    - ASICs “copy” so don’t care.
  - Thus: haven’t needed this during implementation.
- Open source implementation
  - VPP/DPDK implementation to be published in 2020
  - Congestion Control
  - IKEv2
- Open to collaboration/interoperability testing.

Moving Forward

- Any remaining comments?
- Ready for WGLC?

Questions and Comments

Backup Slides

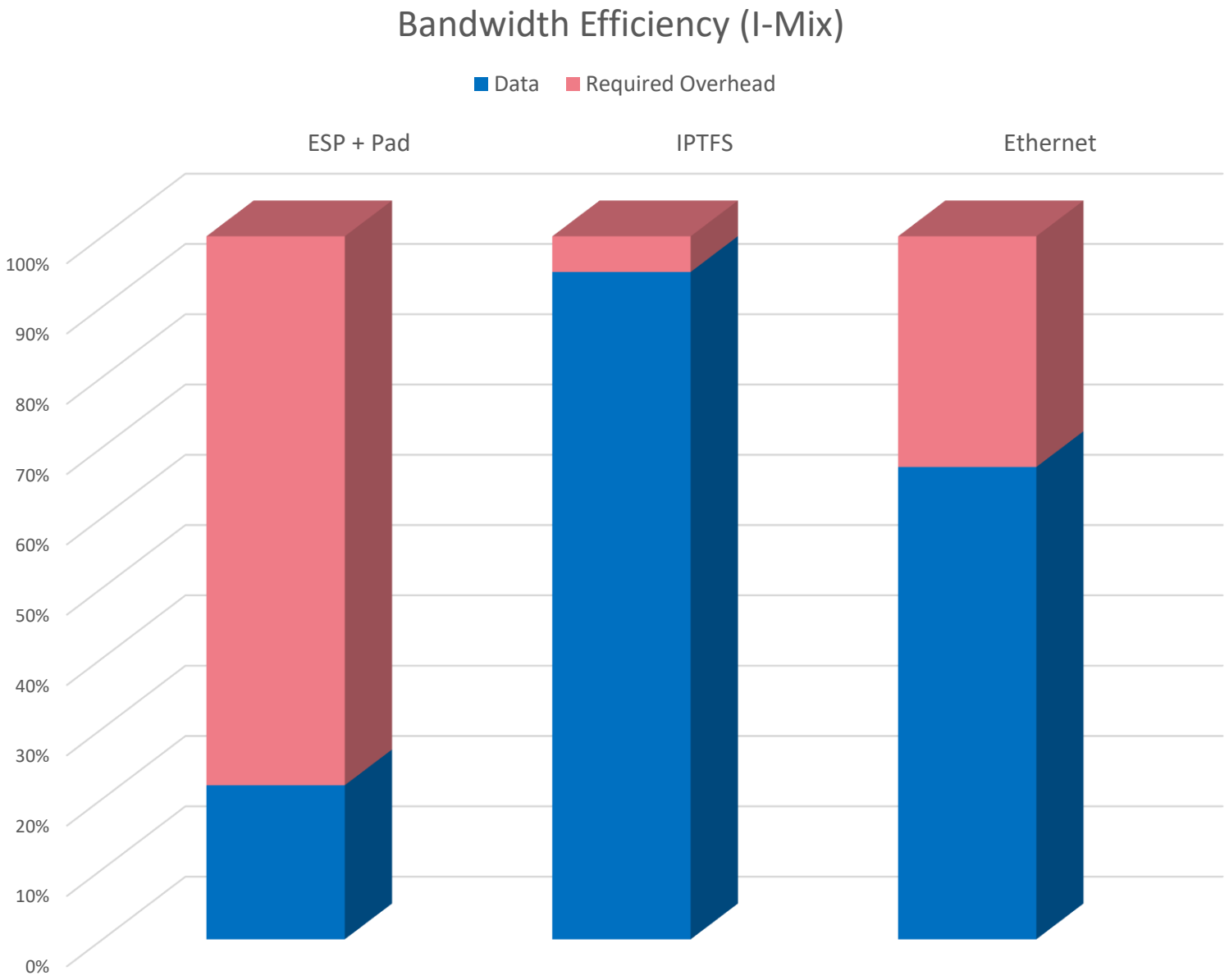
Transport Mode

- Motivation is common GRE/IPsec-Transport Use
- Some interest in generic transport mode.
- What IP header fields to support
  - Simple
    - No fields – GRE Support
      - If the packet header is different then the last, pad current IPTFS out and start new one
      - If is inefficient due to frequent header differences, then use tunnel mode.
    - All Fields
      - IP header replicated inside payload for each packet
      - Similar to tunnel mode, but less efficient.
  - Complex
    - IP Header compression Ideas (deviations, etc)
      - Complex solution in need of a problem?
- Enough separable work to publish as a separate document.

Comparison Data

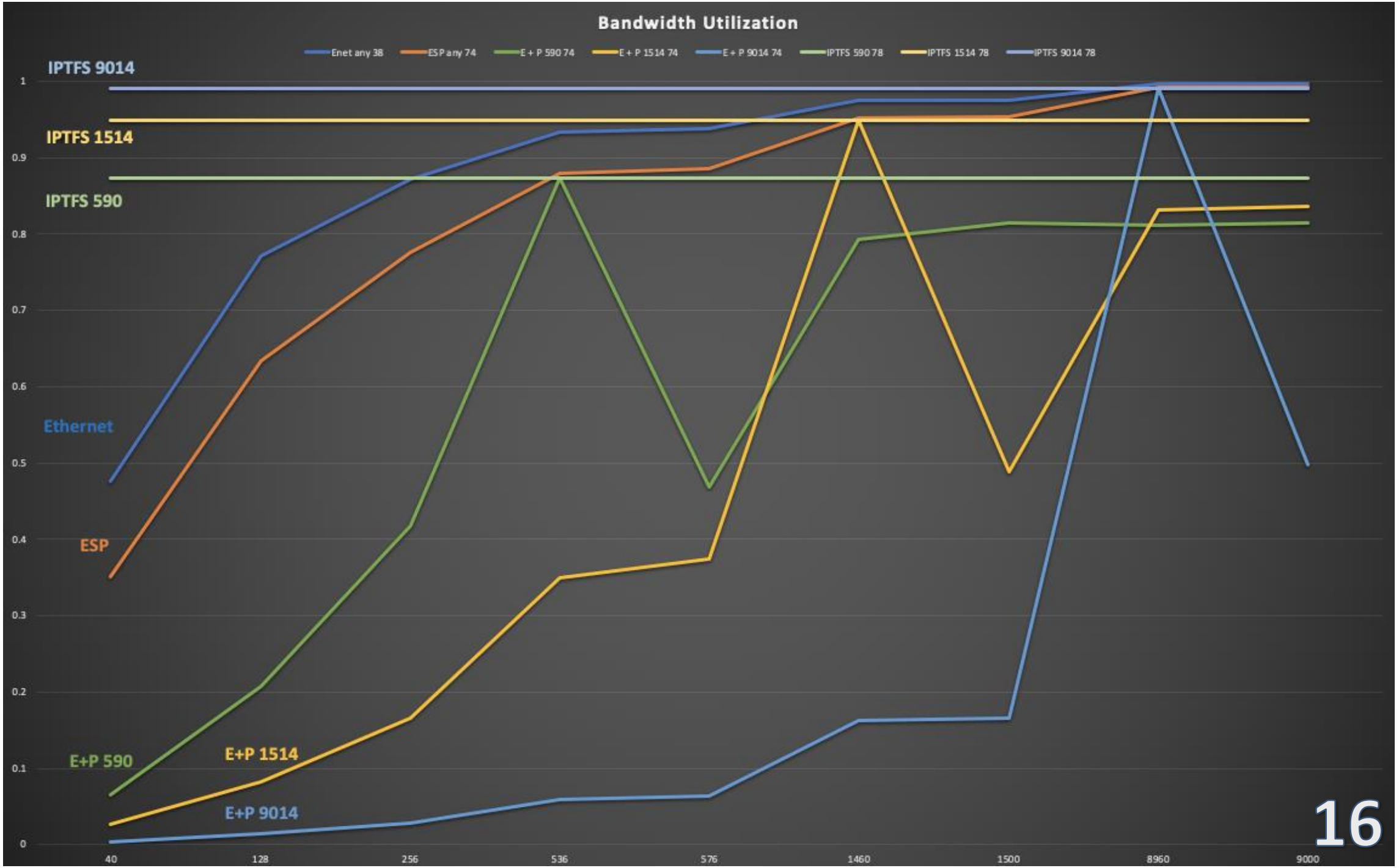
Why is this Needed?

- Current Solution: ESP + Padding 1:1
- Not Deployable.



Solution Cost (I-Mix)

	ESP + Pad	IPTFS	Enet
Bandwidth Used	1Gb	1Gb	1Gb
I-Mix Throughput	219Mb	943Mb	672Mb


Bandwidth Utilization
Enet any 38
ESP any 74
E + P 590 74
E + P 1514 74
E + P 9014 74
IPTFS 590 78
IPTFS 1514 78
IPTFS 9014 78
IPTFS 9014
IPTFS 1514
IPTFS 590
Ethernet
ESP
E+P 590
E+P 1514
E+P 9014
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
40
128
256
536
576
1460
1500
8960
9000

Overhead Comparison in Octets

Type	ESP+Pad	ESP+Pad	ESP+Pad	IP-TFS	IP-TFS	IP-TFS
L3 MTU	576	1500	9000	576	1500	9000
PSize	540	1464	8964	536	1460	8960
40	500	1424	8924	3.0	1.1	0.2
128	412	1336	8836	9.6	3.5	0.6
256	284	1208	8708	19.1	7.0	1.1
536	4	928	8428	40.0	14.7	2.4
576	576	888	8388	43.0	15.8	2.6
1460	268	4	7504	109.0	40.0	6.5
1500	228	1500	7464	111.9	41.1	6.7
8960	1408	1540	4	668.7	245.5	40.0
9000	1368	1500	9000	671.6	246.6	40.2

Overhead as Percentage of Inner Packet

Type	ESP+Pad	ESP+Pad	ESP+Pad	IP-TFS	IP-TFS	IP-TFS
MTU	576	1500	9000	576	1500	9000
PSize	540	1464	8964	536	1460	8960
40	1250.0%	3560.0%	22310.0%	7.46%	2.74%	0.45%
128	321.9%	1043.8%	6903.1%	7.46%	2.74%	0.45%
256	110.9%	471.9%	3401.6%	7.46%	2.74%	0.45%
536	0.7%	173.1%	1572.4%	7.46%	2.74%	0.45%
576	100.0%	154.2%	1456.2%	7.46%	2.74%	0.45%
1460	18.4%	0.3%	514.0%	7.46%	2.74%	0.45%
1500	15.2%	100.0%	497.6%	7.46%	2.74%	0.45%
8960	15.7%	17.2%	0.0%	7.46%	2.74%	0.45%
9000	15.2%	16.7%	100.0%	7.46%	2.74%	0.45%

Bandwidth Utilization over Ethernet

	Enet	ESP	E + P	E + P	E + P	IPTFS	IPTFS	IPTFS
	any	any	590	1514	9014	590	1514	9014
Size	38	74	74	74	74	78	78	78
40	47.6%	35.1%	6.5%	2.6%	0.4%	87.3%	94.9%	99.1%
128	77.1%	63.4%	20.8%	8.3%	1.4%	87.3%	94.9%	99.1%
256	87.1%	77.6%	41.7%	16.6%	2.8%	87.3%	94.9%	99.1%
536	93.4%	87.9%	87.3%	34.9%	5.9%	87.3%	94.9%	99.1%
576	93.8%	88.6%	46.9%	37.5%	6.4%	87.3%	94.9%	99.1%
1460	97.5%	95.2%	79.3%	94.9%	16.2%	87.3%	94.9%	99.1%
1500	97.5%	95.3%	81.4%	48.8%	16.6%	87.3%	94.9%	99.1%
8960	99.6%	99.2%	81.1%	83.2%	99.1%	87.3%	94.9%	99.1%
9000	99.6%	99.2%	81.4%	83.6%	49.8%	87.3%	94.9%	99.1%

Latency

- Latency values seem very similar
- IP-TFS values represent max latency
- IP-TFS provides for constant high bandwidth
- ESP + padding value represents min latency
- ESP + padding often greatly reduces available bandwidth.

	ESP+Pad 1500	ESP+Pad 9000	IP-TFS 1500	IP-TFS 9000
40	1.14 us	7.14 us	1.17 us	7.17 us
128	1.07 us	7.07 us	1.10 us	7.10 us
256	0.97 us	6.97 us	1.00 us	7.00 us
536	0.74 us	6.74 us	0.77 us	6.77 us
576	0.71 us	6.71 us	0.74 us	6.74 us
1460	0.00 us	6.00 us	0.04 us	6.04 us
1500	1.20 us	5.97 us	0.00 us	6.00 us