Accelerated IPsec-VPN Communication with T6

Chelsio T6 vs. Intel AES-NI

Executive Summary

Chelsio Crypto Accelerator is a co-processor designed specifically to perform computationally intensive cryptographic operations more efficiently than general-purpose CPUs. Servers with system load, comprising of cryptographic operations, see great performance improvement by offloading crypto operations on to the Chelsio Unified Wire adapter. Chelsio's solution uses the standard crypto API framework provided by the operating system and enables the offloading of crypto operations on to the adapter.

This paper highlights Chelsio T6 Unified Wire adapters' unique accelerating capabilities for secure IPsec-based VPN connections by comparing its bandwidth and CPU usage with Intel AES-NI. T6 provides consistently higher throughput across the range of I/O sizes compared to Intel AES-NI. Furthermore, CPU usage was less than 5% across the board. Chelsio T6's IPsec-VPN solution provides enterprises with secure remote connection to access corporate applications and resources without sacrificing on performance and speed.

Chelsio VPN Acceleration

The Terminator 6 (T6) ASIC from Chelsio Communications, Inc. is a sixth generation, high performance 1/10/25/40/50/100Gbps, unified wire engine which offers crypto offload capability for AES and SHA variants. Internet Protocol Security (IPsec) is an end-to-end security scheme that provides protocols to ensure the authenticity, privacy and integrity of data in transit. Virtual Private Network (VPN) is a network connection that secures traffic between locations. Chelsio solution provides an accelerated IPsec-VPN tunnel which is well suited for site-to-site security over WAN.

Chelsio crypto accelerator secures data using AES (Advanced Encryption Standard) - the strongest encryption algorithm available. Encryption and decryption processing for IPsec is offloaded on to the T6 adapter freeing CPU resources for other tasks. Chelsio crypto driver registers with the kernel crypto framework with high priority and ensures that encryption request is offloaded and processed by T6. IPsec protocol integrated in the kernel calls the crypto API framework which transforms the API into Chelsio supported crypto routines. The data is encrypted and decrypted in the loopback mode for both Tx and Rx paths.

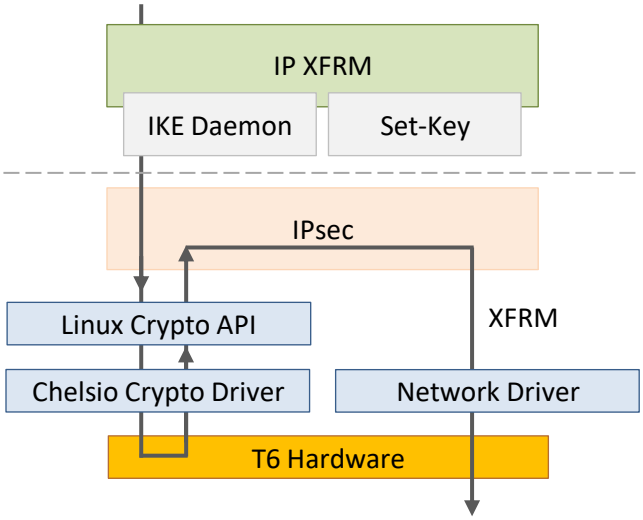

Figure 1 – T6 IPsec Acceleration

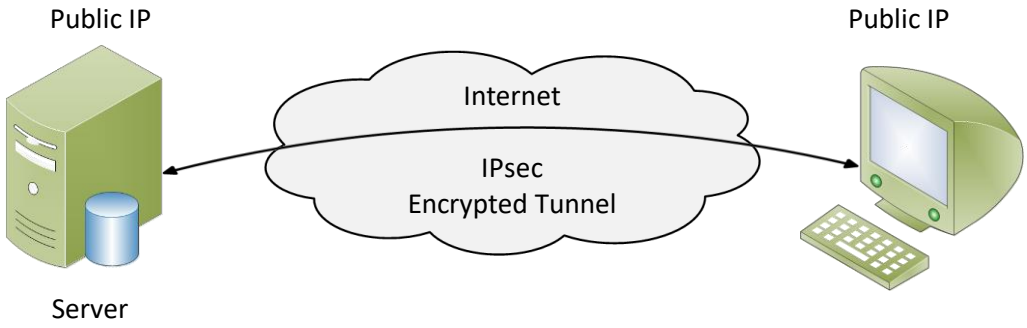

Figure 2 - IPsec Transport Mode

Test Results

The following graph compares the throughput and CPU Usage of Chelsio crypto (offload) and Intel AES-NI modes using the **iperf** tool.

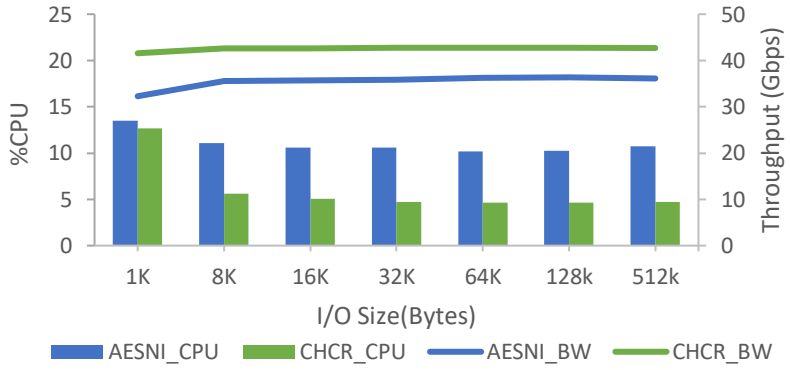

Figure 3 – Throughput and CPU % of Chelsio crypto vs. Intel AES-NI

Chelsio crypto solution delivers upto 15% higher throughput than Intel AES-NI across the range of I/O sizes. Also, evident from the graph is Chelsio's improved efficiency which is 50% less than Intel AES-NI from 8KB I/O size.

Test Configuration

The setup consists of two machines connected back-to-back using a single 100GbE port- a Server and a Client. Each system was configured with 2 Intel Xeon CPU E5-2687W v4 24-core processors clocked at 3.00GHz (HT enabled), 128GB of RAM and RHEL 7.3 operating system (kernel 4.9.13). Chelsio T62100-CR adapter was installed in each system and configured with Co-processor Driver v1.0.0.0. iproute rpm package was installed. MTU of 9000B was used.

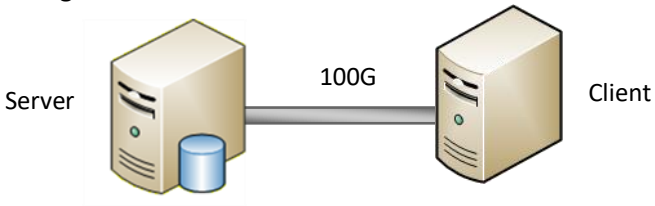

Figure 4 – Back to Back topology

Test Configuration

i. Install the driver package.

```
[root@host ~]# ./install.sh -i
```

ii. Reboot the machine into newly installed kernel.

iii. Install Chelsio Crypto accelerator driver.

```
[root@host ~]# ./install.sh -d
```

iv. Chelsio network driver was loaded on Server and Client.

```
[root@host ~]# modprobe cxgb4
```

v. Respective drivers were loaded for Chelsio crypto and Intel AES-NI modes.

Chelsio crypto:

```
[root@host ~]# modprobe -v chcr
```

Intel AES-NI:

```
[root@host ~]# modprobe aesni_intel
```

vi. IP alias was configured on Chelsio interfaces.

```
[root@host~ ]#  for i in `seq 1 8`;do ifconfig ethX:$i $i.0.0.2/24 up;done
```

vii. CPU affinity was set.

```
[root@host~ ]# t4_perftune.sh -Q nic
[root@host~ ]# t4_perftune.sh -Q crypto
```

viii. The following *sysctl* parameters were set:

```
sysctl -w net.ipv4.tcp_timestamps=0
sysctl -w net.core.netdev_max_backlog=250000
sysctl -w net.core.rmem_max=4194304
sysctl -w net.core.wmem_max=4194304
sysctl -w net.core.rmem_default=4194304
sysctl -w net.core.wmem_default=4194304
sysctl -w net.ipv4.tcp_rmem="4096        87380   4194304"
sysctl -w net.ipv4.tcp_wmem="4096        16384   4194304"
```

ix. ip-xfrm was configured using below scripts on Server and Client.

Server

```
for i in `seq 1 8`
do
local_ip=$i.0.0.1
remote_ip=$i.0.0.2
ip xfrm state add src $local_ip dst $remote_ip proto esp spi 0x53fa1fdd reqid
16386 mode transport  aead "rfc4106(gcm(aes))"
0x010203047aeaca3f87d060a12f4a4487d5a5c335 96 sel src 0.0.0.0/0 dst 0.0.0.0/0
ip xfrm state add src $remote_ip dst $local_ip proto esp spi 0x53fa1fde reqid
16385 mode transport aead "rfc4106(gcm(aes))"
0x010203047aeaca3f87d060a12f4a4487d5a5c335 96 sel src 0.0.0.0/0 dst 0.0.0.0/0
```

```
ip xfrm policy add src $local_ip dst $remote_ip dir out tmpl src $local_ip dst
$remote_ip proto esp reqid 16386  mode transport
ip xfrm policy add src $local_ip dst $remote_ip dir fwd tmpl src $local_ip dst
$remote_ip proto esp reqid 16385 mode transport
ip xfrm policy add src $local_ip dst $remote_ip  dir in tmpl src $local_ip dst
$remote_ip proto esp reqid 16385 mode transport
done
```

Client

```
for i in `seq 1 8`
do
local_ip=$i.0.0.2
remote_ip=$i.0.0.1
ip xfrm state add src $remote_ip dst $local_ip proto esp spi 0x53fa1fdd reqid
16386 mode transport aead "rfc4106(gcm(aes))"
0x010203047aeaca3f87d060a12f4a4487d5a5c335 96 sel src 0.0.0.0/0 dst 0.0.0.0/0
ip xfrm state add src $local_ip dst $remote_ip proto esp spi 0x53fa1fde reqid
16385 mode transport aead "rfc4106(gcm(aes))"
0x010203047aeaca3f87d060a12f4a4487d5a5c335 96 sel src 0.0.0.0/0 dst 0.0.0.0/0
ip xfrm policy add src $local_ip dst $remote_ip dir out tmpl src $local_ip dst
$remote_ip proto esp reqid 16385  mode transport
ip xfrm policy add src $local_ip dst $remote_ip dir fwd tmpl src $local_ip dst
$remote_ip proto esp reqid 16386 mode transport
ip xfrm policy add src $local_ip dst $remote_ip dir in tmpl src $local_ip dst
$remote_ip proto esp reqid 16386 mode transport
done
```

x. The xfrm policies were verified:

```
[root@host~ ]# ip xfrm state list
```

xi. iperf servers with 8 different IP alias were started on Server.

```
[root@host~ ]# for i in `seq 1 8`; do taskset -c 0-23 iperf -s -w 512k  -p 500$i
& done
```

xii. Connection to the listening servers was established from the client.

```
[root@host~ ]# for i in `seq 1 8`; do taskset -c $i iperf -c $i.0.0.1 -p 500$i -w
512K -l <I/O size> -t 30 & done
```

Conclusion

This paper presented performance comparison of Chelsio's T6 IPsec-VPN solution and Intel's AES-NI using T62100-CR adapter in Linux. Chelsio outperformed Intel AES-NI with a consistently higher throughput across the range of study. In addition, Chelsio's CPU usage was half of Intel's indicative of a more efficient processing path. Chelsio's T6 low-cost IPsec-based VPN solution provides data transfer with high accuracy and speed without affecting integrity and confidentiality.

Related Links

Apache TLS/SSL Acceleration at 100GbE
Disk Encryption with 100GbE Crypto Accelerator
T6 100GbE Crypto Offload Performance
Chelsio Terminator 6 ASIC 100GE Crypto Offload
The Chelsio Terminator 6 ASIC
Chelsio T6 Crypto Offload Video