A PNA proposal to standardize checksum externs

Proposal to standardize checksum externs

1. DPU needs to perform checksum verification of incoming packet and compute checksum for outgoing packet
2. L3, L4 Checksum verification
    1. Verify more than one layer of L3 and L4 checksums.
    2. Provide checksum verification result.
    3. Skip verification of zero checksum value. (For certain packet encapsulation, it's fine to have zero UDP checksum)
3. L3, L4 Checksum computation
    1. Compute more than one layer of L3 and L4 checksums
    2. Include computed checksum of inner packet layer/s into outer layer's packet checksum.
4. Other checksum computation
    1. Compute complete checksum for Linux kernel to use (Rx Packet)

Proposal to standardize checksum externs

1. Unified checksum extern class and methods for checksum verification and computation
2. Depending upon DPU architecture checksum objects and methods are used in appropriate stages in pipeline. Define methods to trigger verification, obtain verified results and to trigger checksum computation.
3. Generally, checksum fields of a packet is verified inside parser. The extern objects and methods could be used in Parser block.
4. Pipeline may trigger checksum computation. There may be a need for the extern objects and methods to be used inside pipeline or action routines

Extern class and methods

```
extern Checksum16 {
    Checksum16();
    Checksum16(bool csum_zero_skip_verify);
    void validate<F>(in F checksum_destination);
    bit<16> get();
    bit<1> validation_failed();
    void checksum_over<O,L>(O offset, L length);
    void add_fields<L>(L field_list);
    void add(bit<16> const_val);
    void enable_update();
    void disable_update();
}
```

- Extern class and methods. Next few slides will describe each of these methods in detail.

Extern class and constructors

```
extern Checksum16 {
    Checksum16();
    Checksum16(bool csum_zero_skip_verify);
}
```

- Default constructor and a variant constructor with an ability to skip verification of packet checksum with zero value
- Computes 16 bits checksum value.

Extern methods to validate, compute, check verification results

```
extern Checksum16 {
        void validate<F>(in F checksum_destination);
        bit<16> get();
        bit<1> validation_failed();
}
```

- The validate() method allows to verify incoming packets checksum value.
  - checksum_destination : Packet header field where checksum value is present in a packet. This parameter is a header field. (Example: header.ipv4.checksum)
- The get() method computes checksum value. The method returns 16 bits two's complement over the packet data. Data over which checksum is computed is provided using update_len() method [Details in next slide].
- The validation_failed() method allows pipeline to check checksum verification result.
  When validation fails, the method returns 1.

Extern methods to setup checksum bytes

```
extern Checksum16 {
    void checksum_over<O,L>(O offset, L length);
}
```

- The checksum_over() method uses offset into packet and length in bytes. Depending on get() method or verify() method, checksum is computed or verified over the length bytes starting from the offset.

Extern methods to setup checksum bytes

```
extern Checksum16 {
    void add_fields<L>(L field_list);
}
```

- The add_fields() method can be used to provide additional packet header fields that need to be part of checksum verification or computation.
- Example, this method can be put to use in scenarios like L4 checksum that uses partial fields from L3 header as well.

Extern methods to setup checksum bytes

```
extern Checksum16 {
     void add(bit<16> const_val);
}
```

- The add() method can be used to provide compile time constant value.
- Example, this method can be used to supply protocol value when its not at a well known offset in L3 header due to presence of options.

Extern method for handling multi layer packets.

```
extern Checksum16 {
    void include_checksum_result<H>(in H checksum_header);
}
```

- This method allows to include checksum computation results of inner headers into checksum computation being performed on outer headers
- Example : When outer UDP checksum and inner UDP checksum are being computed, include inner UDP checksum result into outer UDP checksum computation.

Extern methods to enable computation

```
extern Checksum16 {
        void enable_update();
}
```

- The enable_update() method allows to trigger checksum computation. An example usage of this method is in p4-action

Extern methods to disable computation

```
extern Checksum16 {
        void disable_update();
}
```

- The disable_update() method allows to disable checksum computation. An example usage of this method is in p4-action. In error / drop scenario, pipeline can disable checksum computation using this method.

Example: Verify ipv4.checksum

```
 Checksum16()          ipv4HdrCsum;
parser  p(packet_in packet, out headers hdr,.. ) {
 bit<16>                        l3_hdr_offset;
 :
 state parse_ipv4 {
   l3_hdr_offset = packet.state_byte_offset();
   packet.extract(hdr.ipv4);
   transition parse_ipv4_checksum;
 }
 state parse_ipv4_checksum {
   bit<16>     ip_header_len;
   ip_header_len = ((bit<16>) hdr.ipv4.ihl << 2);
   ipv4HdrCsum.checksum_over(l3_hdr_offset, ip_header_len);
   ipv4HdrCsum.validate(hdr.ipv4.checksum);
 }
  :
}
```

- Code snippet to show how checksum extern object and methods will be used to verify ip header checksum.
- The result of the checksum verification can be obtained in match action pipeline using ipv4HdrCsum.validation_failed()

Example: Verify L4 checksum (udp or tcp or icmp)

```
 Checksum16()         udpCsum;
parser  p(packet_in packet, out headers hdr,.. ) {
 bit<16>                      l4_hdr_offset;
 :
 state parse_ipv4 {
  l3_hdr_offset = packet.state_byte_offset();
  packet.extract(hdr.ipv4);
  ip_total_len  = hdr.ipv4.totalLen;
  bit <16> ip_header_len = ((bit<16>) hdr.ipv4.ihl << 2);
  l4_len        = ip_total_len - ip_header_len;
  transition parse_udp;
 }
 state parse_udp {
  packet.extract(hdr.udp);
  l4_hdr_offset = packet.state_byte_offset();
  udpCsum.checksum_over(l4_hdr_offset, l4_len);
  udpCsum.add_fields({hdr.ipv4.srcAddr, hdr.ipv4.dstAddr, l4_len});
  udpCsum.add(IP_PROTO_UDP);
  udpCsum.validate(hdr.udp.checksum);
 }

 .
```

- Code snippet to show how checksum extern object and methods will be used to verify udp checksum.
- The result of the checksum verification can be obtained in match action pipeline using udpCsum.validation_failed()

Example: Compute ipv4.checksum

```
Checksum16()          ipv4HdrCsum;
 control deparser (packet_out packet, inout headers hdr,
     control_metadata metadata, .. ) {
   apply {
     packet.emit(hdr.ipv4);
     If (hdr.tcp.valid()) {
        ipv4HdrCsum.checksum_over(hdr.ipv4,
     metadata.csum.ip_hdr_len);
        hdr.ipv4.checksum = ipv4HdrCsum.get();
     }
   }
}


control match-action ( inout headers hdr, control_metadata metadata,
     .. ) {
   action  foo() {
     ipv4HdrCsum.enable_update();
   }
 }
```

- Code snippet to show how checksum extern object and methods will be used to compute ip header checksum.
- The result of the checksum computation can be obtained and set into ipv4 header checksum field using get() method

Example: Compute tcp.checksum

```
Checksum16()        tcpCsum;
 control deparser (packet_out packet, inout headers hdr, control_metadata
     metadata, .. ) {
   apply {
     packet.emit(hdr.tcp);
     If (hdr.tcp.valid()) {
        If (hdr.ipv4.valid()) {
          tcpCsum.add_fields({hdr.ipv4.srcAddr,
          hdr.ipv4.dstAddr, metadata.csum.tcp_len});
        }
        If (hdr.ipv6.valid()) {
           tcpCsum.add_fields({hdr.ipv6.srcAddr,
           hdr.ipv6.dstAddr, metadata.csum.tcp_len});
        }
        tcpCsum.add(IP_PROTO_TCP);
        tcpCsum.checksum_over(hdr.tcp, metadata.csum.tcp_len);
        tcpCsum.enable_update();
        hdr.tcp.checksum = tcpCsum.get();
     }
 }
```

- Code snippet to show how checksum extern object and methods will be used to compute tcp checksum.
- The result of the checksum computation can be obtained and set into tcp header checksum field using get() method.
- Since ipv6 next header field can be at non fixed offset from the start of the header (depends on ipv6 options), for tcp checksum with v6 L3 header, TCP protocol constant value is added as pseudo header field.

Example: Compute outer udp checksum, inner tcp

```
Checksum16()        outerUdpCsum, tcpCsum;
control deparser (packet_out packet, inout headers hdr, control_metadata
    metadata, .. ) {
  apply {
    packet.emit(hdr.outer_udp);
    If (hdr.outer_udp.valid()) {
      If (hdr.ipv4.valid()) {
          outerUdpCsum.add_fields({hdr.ipv4.srcAddr,
          hdr.ipv4.dstAddr, metadata.csum.outer_udp_len});
        }
        If (hdr.ipv6.valid()) {
          outerUdpCsum.add_fields({hdr.ipv6.srcAddr,
          hdr.ipv6.dstAddr, metadata.csum.outer_udp_len});
        }
        If (hdr.tcp.valid()) {
          // Outer Udp, Inner TCP
          outerUdpCsum.add_fields({hdr.tcp.checksum});
       }
       If (hdr.tcp.valid()) {
          // Outer Udp, Inner udp
          outerUdpCsum.add_fields({hdr.inner_udp.checksum});
       }
       :
       :
```

```
       :
       :
       outerUdpCsum.add(IP_PROTO_UDP);
       outerUdpCsum.checksum_over(hdr.outer_udp,
                                  metadata.csum.outer_udp_len);
       outerUdpCsum.enable_update();
       hdr.outerUdp.checksum = outerUdpCsum.get();
    }
    packet.emit(hdr.tcp);
    If (hdr.tcp.valid()) {
       If (hdr.ipv4.valid()) {
          tcpCsum.add_fields({hdr.ipv4.srcAddr,
          hdr.ipv4.dstAddr, metadata.csum.tcp_len});
        }
        If (hdr.ipv6.valid()) {
          tcpCsum.add_fields({hdr.ipv6.srcAddr,
          hdr.ipv6.dstAddr, metadata.csum.tcp_len});
        }
       tcpCsum.add(IP_PROTO_TCP);
       tcpCsum.checksum_over(hdr.tcp, metadata.csum.tcp_len);
       tcpCsum.enable_update();
       hdr.tcp.checksum = tcpCsum.get();
    }
  }
```