UDP Options Implementation Experience

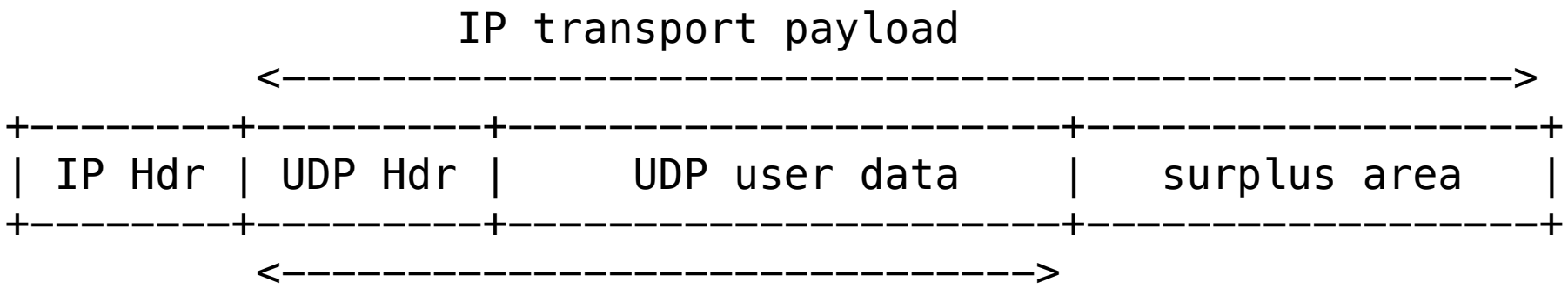
draft-ietf-tsvwg-udp-options-02

Tom Jones

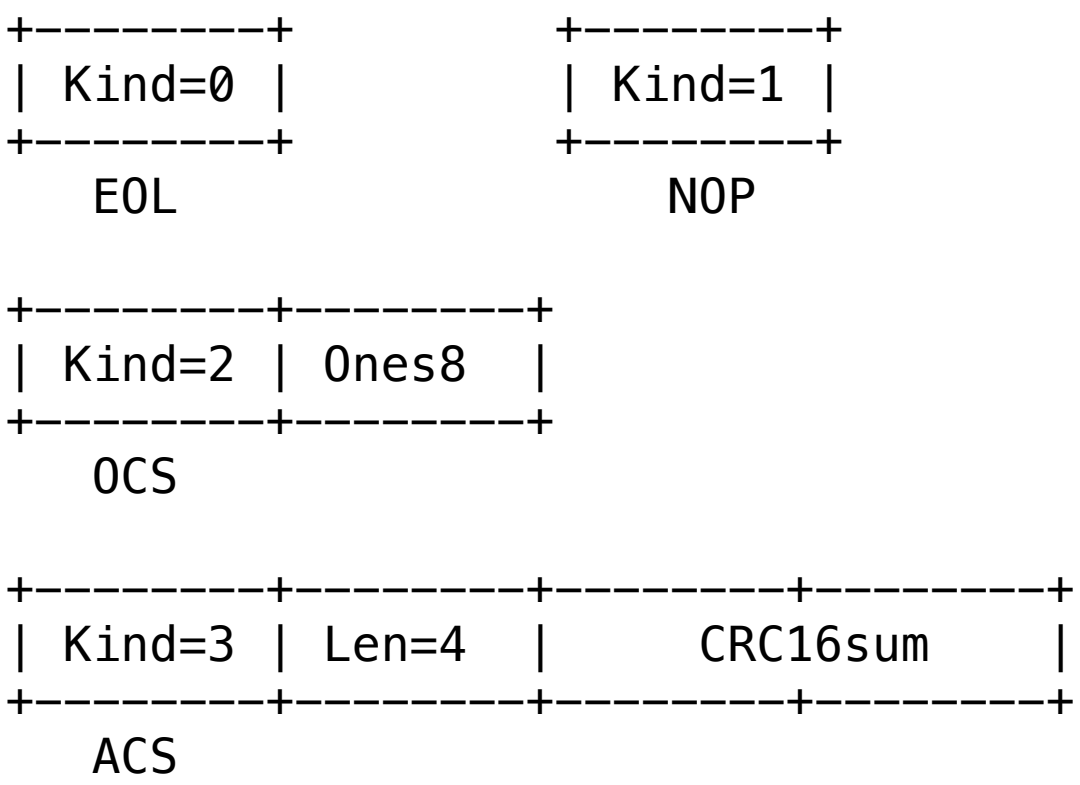
tom@erg.abdn.ac.uk

NEAT is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement no. 644334.

UDP Option Area



UDP Option TLV



UDP Options

Kind	Length	Meaning
0*	–	End of Options List (EOL)
1*	–	No operation (NOP)
2*	2	Option checksum (OCS)
3	4	Alternate checksum (ACS)
4	4	Lite (LITE)
5	4	Maximum segment size (MSS)
6	10	Timestamps (TIME)
7	12	Fragmentation (FRAG)
8	(varies)	Authentication and Encryption (AE)
9–126	(varies)	UNASSIGNED (assignable by IANA)
127–253		RESERVED
254	N(>=4)	RFC 3692-style experiments (EXP)
255		RESERVED

UDP Options

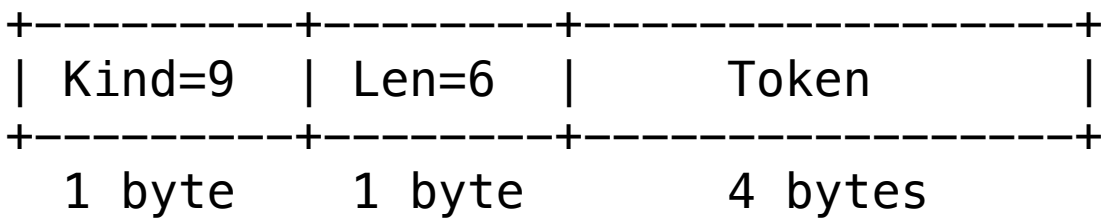
Kind	Length	Meaning
0*	–	End of Options List (EOL)
1*	–	No operation (NOP)
2*	2	Option checksum (OCS)
3	4	Alternate checksum (ACS)
4	4	Lite (LITE)
5	4	Maximum segment size (MSS)
6	10	Timestamps (TIME)
7	12	Fragmentation (FRAG)
8	(varies)	Authentication and Encryption (AE)
9–126	(varies)	UNASSIGNED (assignable by IANA)
127–253		RESERVED
254	N(>=4)	RFC 3692-style experiments (EXP)
255		RESERVED

Implementation Status

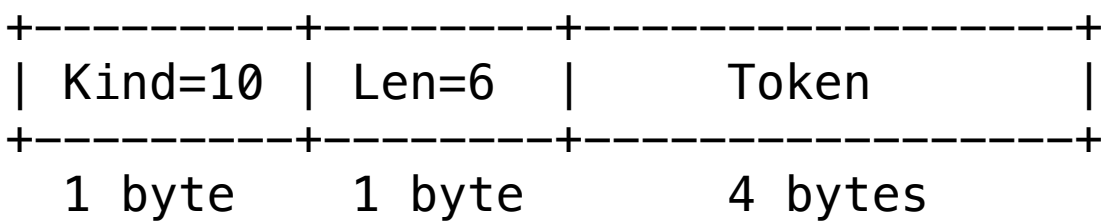
- Implementation for FreeBSD
  - https://github.com/uoaerg/freebsd
    - branch udpoptions-ietf101
- Wireshark dissector
- PacketDrill Tests

New Options to support DPLPMTUD

- Add two New Options
  - Echo Request



  - Echo Response

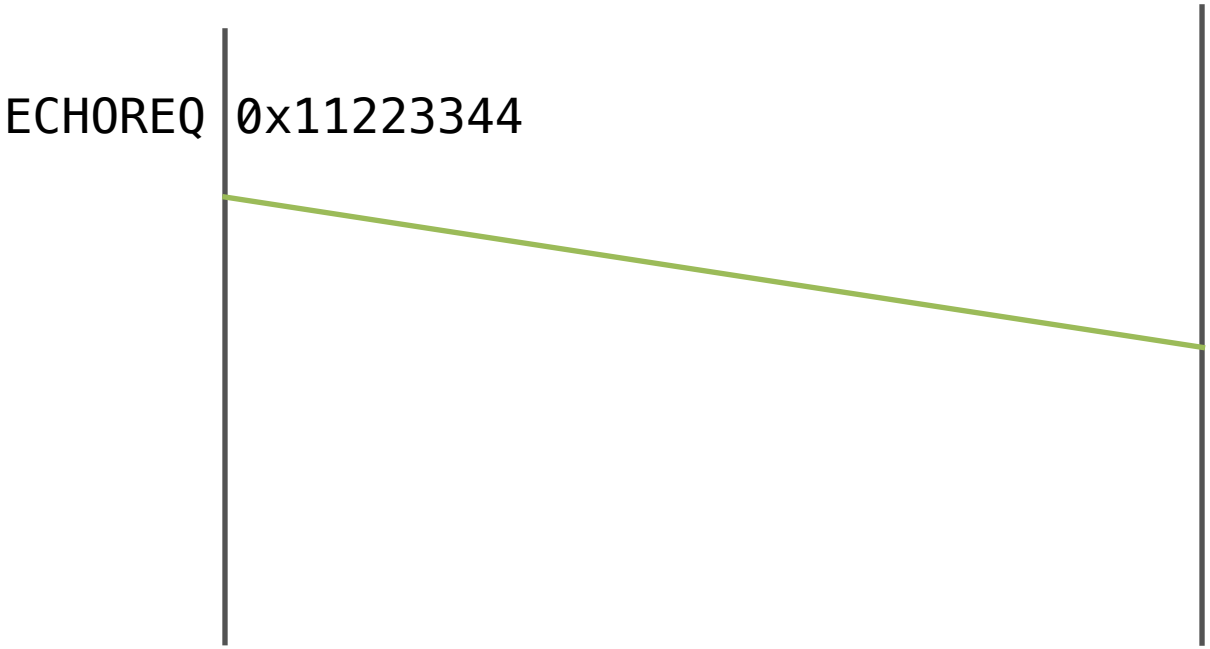


- Probe: Echo request with entire padding packet

Why do we need two options

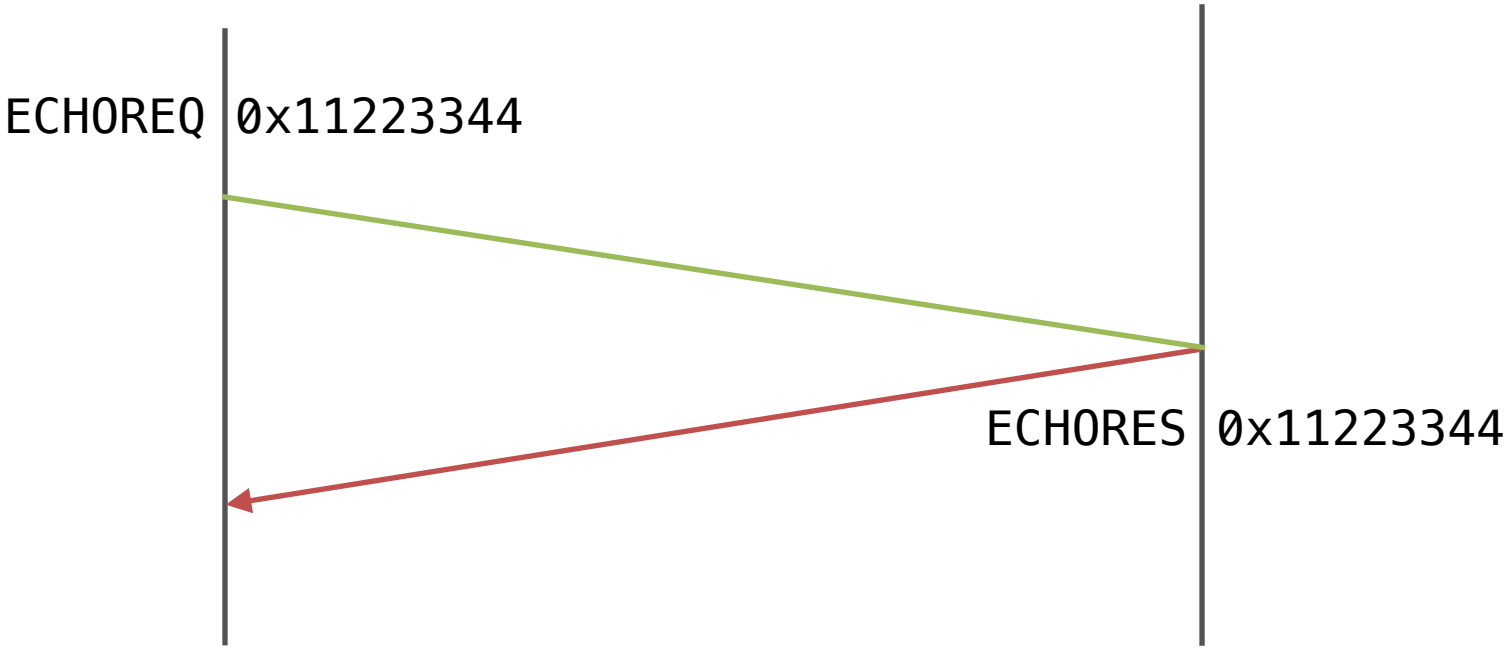
- Each data direction is a separate unidirectional stream
- Use echo request and response in each direction
  - Uses:
    - To verify UDP Options is supported on the path
    - As a connectivity check
    - To verify PMTU Probe is received
    - Could verify remote receives a specific “option”

Why do we need two options



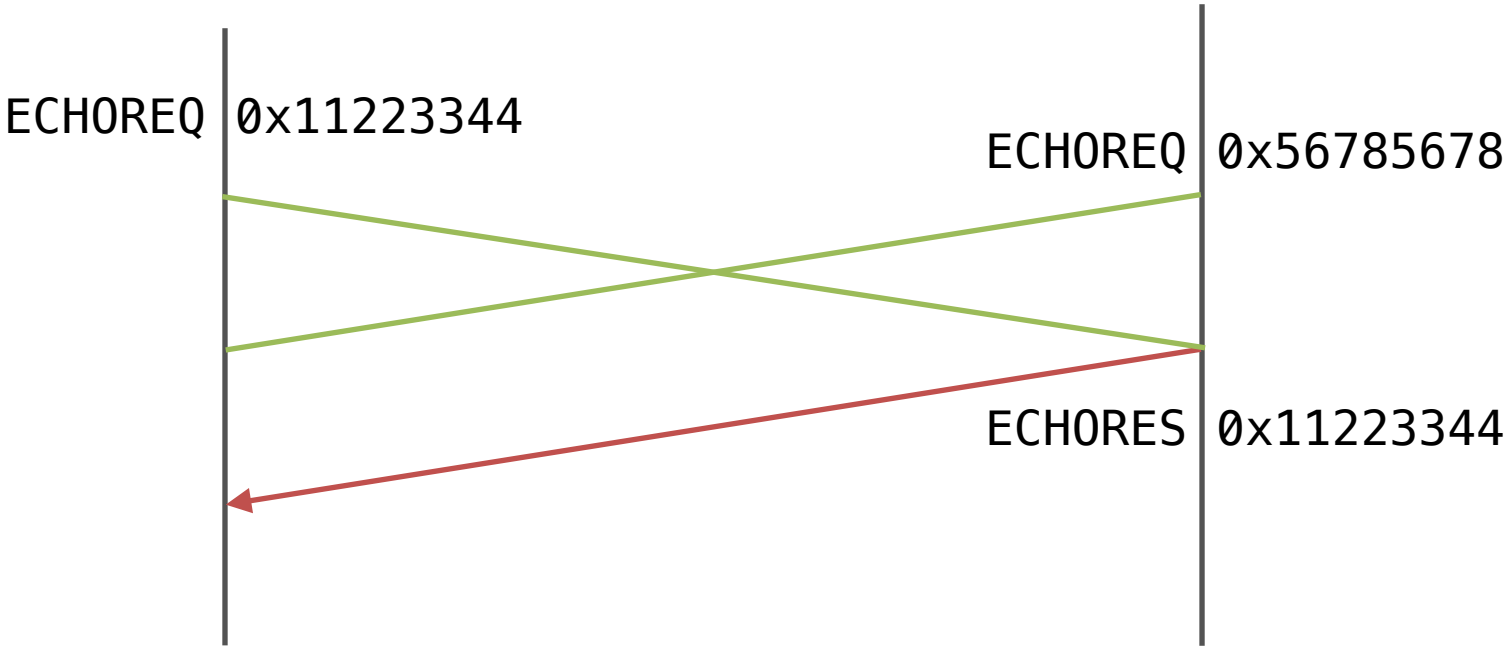
- Each data direction is a separate unidirectional stream
- Use echo request and response in each direction
  - Uses:
    - To verify UDP Options is supported on the path
    - As a connectivity check
    - To verify PMTU Probe is received
    - Could verify remote receives a specific “option”

Why do we need two options



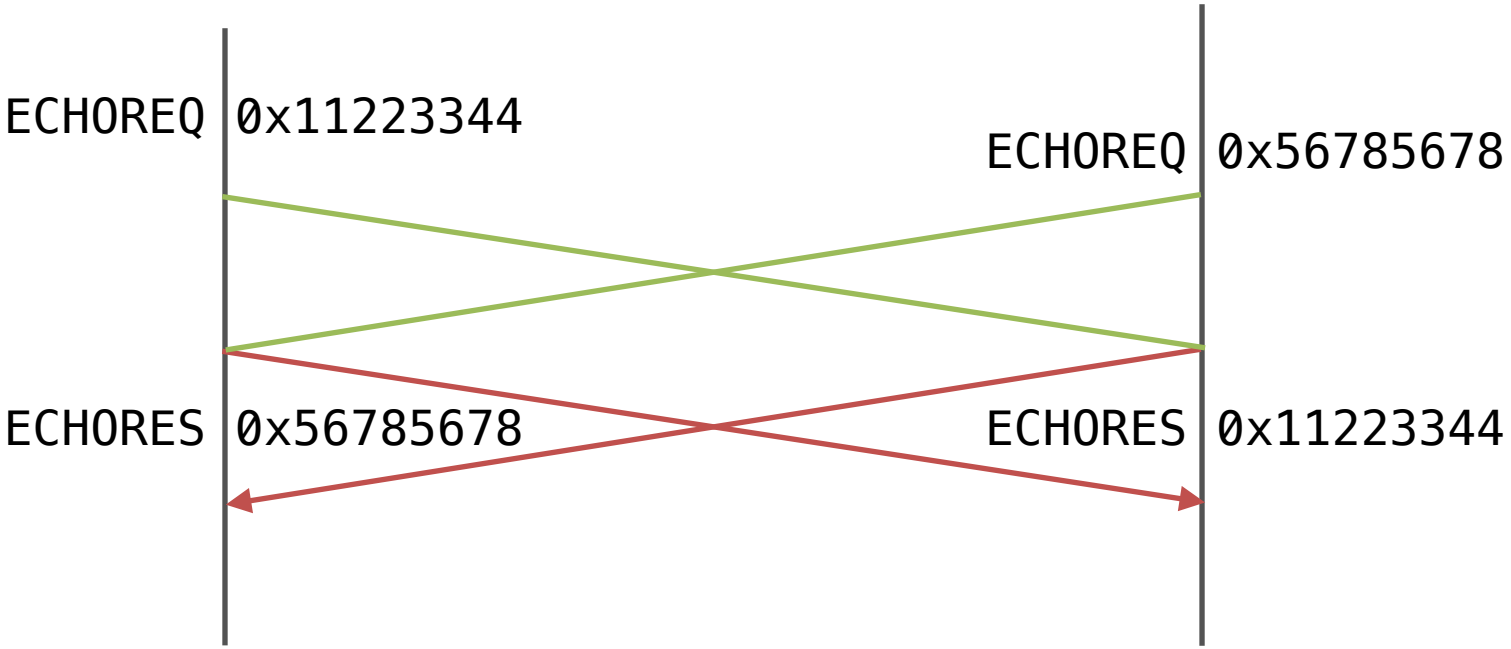
- Each data direction is a separate unidirectional stream
- Use echo request and response in each direction
  - Uses:
    - To verify UDP Options is supported on the path
    - As a connectivity check
    - To verify PMTU Probe is received
    - Could verify remote receives a specific “option”

Why do we need two options



- Each data direction is a separate unidirectional stream
- Use echo request and response in each direction
  - Uses:
    - To verify UDP Options is supported on the path
    - As a connectivity check
    - To verify PMTU Probe is received
    - Could verify remote receives a specific “option”

Why do we need two options



- Each data direction is a separate unidirectional stream
- Use echo request and response in each direction
  - Uses:
    - To verify UDP Options is supported on the path
    - As a connectivity check
    - To verify PMTU Probe is received
    - Could verify remote receives a specific “option”

Problem with OCS

"I was proposing that the current OCS option be discarded, and replaced with a fixed length 16 bit checksum field at the start of the surplus area…

- Derek Fawcus

Problem with OCS

"I was proposing that the current OCS option be discarded, and replaced with a fixed length 16 bit checksum field at the start of the surplus area…

So I am again proposing:

1. Remove the OCS option.
2. Add a fixed length 16 bit checksum field at a known location (The start of the surplus area).
3. Potentially allow one to store 0x0000 in that field to indicate the options themselves are not protected by a checksum."

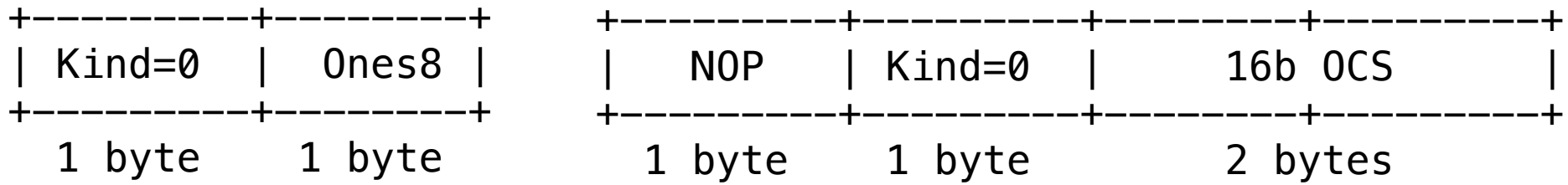
- Derek Fawcus

Problem with OCS

- Option space may need to start with padding
- OCS cannot be fixed at start of option space

Suggestion:

- Redefine EOL as (EOL+OCS)
- Always placed at end of packet



- Makes option layout easier
- Reduces minimum option space Len by 1
- Option space becomes:
  - OPT,OPT,(***NOP***,**EOL+OCS**)

Next Steps

- My implementation will be done in June
- Users?
- Other implementations?

Acknowledgement

NEAT is funded by the European Union's Horizon 2020 research and innovation programme under grant agreement no. 644334.

Possible end host breakage

```
void
in_delayed_cksum(struct mbuf *m)
{
    struct ip *ip;
    uint16_t csum, offset, ip_len;

    ip = mtod(m, struct ip *);
    offset = ip->ip_hl << 2 ;
    ip_len = ntohs(ip->ip_len);
    csum = in_cksum_skip(m, ip_len, offset);
    if (m->m_pkthdr.csum_flags & CSUM_UDP && csum == 0)
        csum = 0xffff;
    offset += m->m_pkthdr.csum_data;    /* checksum offset */
    /* find the mbuf in the chain where the checksum starts*/
    while ((m != NULL) && (offset >= m->m_len)) {
        offset -= m->m_len;
        m = m->m_next;
    }
    *(u_short *)(m->m_data + offset) = csum;
}
```