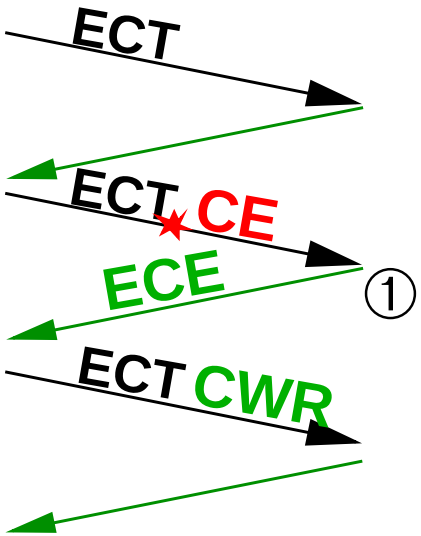
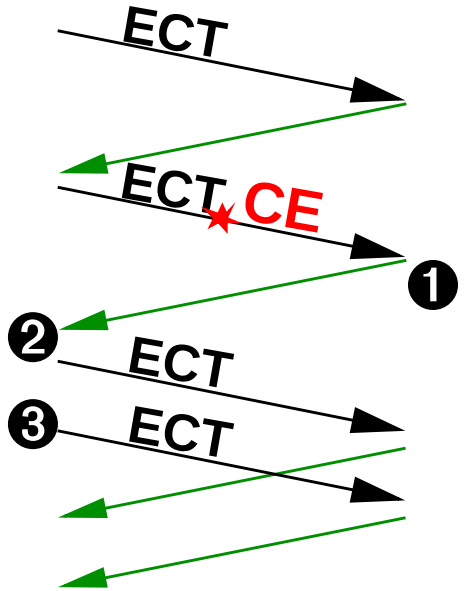
# Robust ECN Signaling with Nonces

David Wetherall, David Ely, and Neil Spring
University of Washington
50th IETF
March, 2001

# Problem

Bugs and misbehavior may hide ECN marks from the sender




① ECN properly echoed

**ECT** – ECN Capable Transport
**CE** – Congestion Experienced
**CWR** – Congestion Window Reduced
**ECE** – ECN-Echo

❶ CE improperly hidden

❷ Sender infers no congestion

❸ Then sends too fast

# Motivation

[SCWA99] lists how and why receivers can fool senders

- Ack division, DupACK spoofing
- Challenging to implement
- Easy to protect against at server

Receivers can’t hide drops if they want data.

Receivers can hide ECN marks and still get data.

Want robust ECN mechanism

- Don’t assume senders trust receivers

# How to hide congestion signals

linux-2.4.0/include/net/tcp_ecn.h

Normal:

```
51: static __inline__ void
52: TCP_ECN_send(...)
...
67:     if (tp->ecn_flags & TCP_ECN_DEMAND_CWR)
68:         skb->h.th->ece = 1;
```

Misbehaving:

```
68:         skb->h.th->ece = 0;
```

# ECN nonce review

One-bit random nonce sent with each packet

- Using same IP header bits as ECT/CE

Nonce is erased to signal congestion

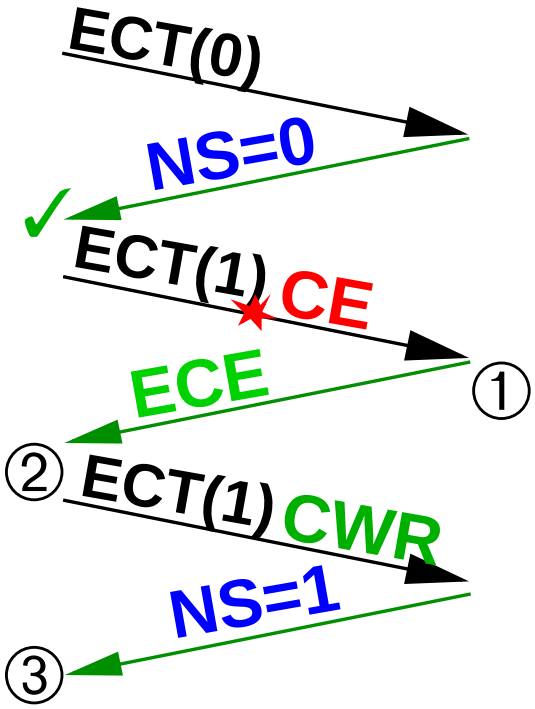
Sum (parity) of nonces returned with each ack

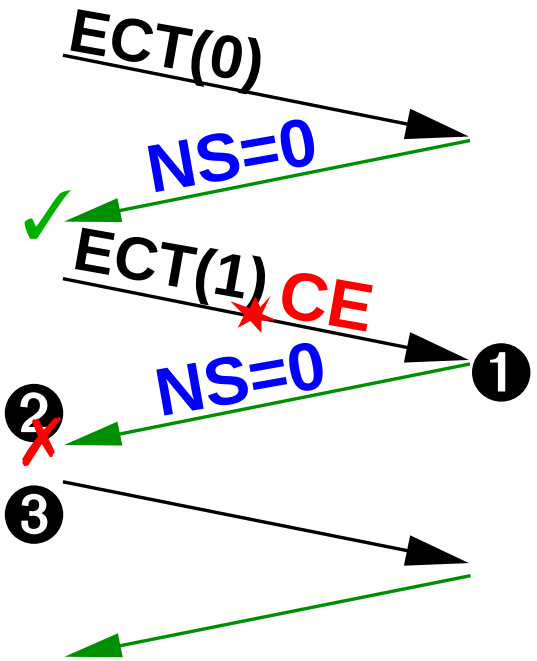
- A new bit from TCP’s reserved field

Sender verifies clear ECN-echo with nonce sum:

- Incorrect sum implies failure
- Should disable ECN and reduce sending rate

# Visual ECN nonce review




① ECN properly echoed

② Nonce sum (**NS**) ignored

③ Synch. **NS** after **CWR**

❶ CE improperly hidden

❷ Guessed **NS** is wrong

❸ Sender disables ECN

# TCP processing state

Receiver stores

- Nonce bit for each out of order packet
- Current nonce sum

Sender stores

- Nonce sum expected for each unack’d packet
- Synchronization offset bit

Packet adds TCP header bit for nonce sum:

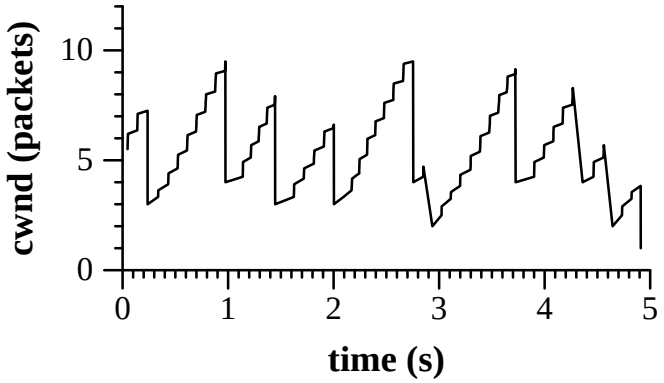
| 4 bit header length | reserved (3 bits) | NS | CWR | ECE | URG | ACK | PSH | RST | SYN | FIN |
|---|---|---|---|---|---|---|---|---|---|---|

# Penalty

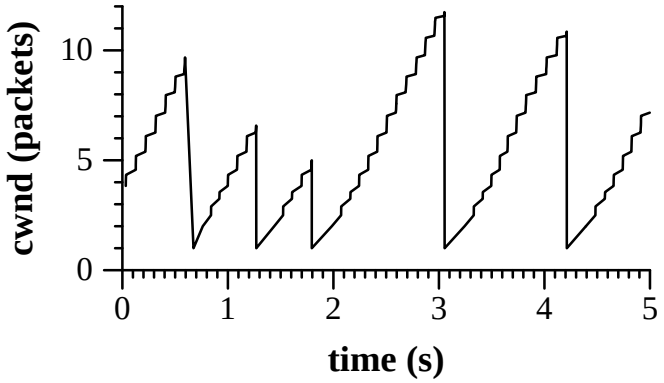
Disabling ECN is not sufficient incentive to behave

Set cwnd = ssthresh = 1

- Intended to negate gains from misbehavior




Normal cwnd sawtooth




cwnd=1 on misbehavior

(but not disabling ECN)

# Alternative approaches

Test correct operation by setting CE bit at sender.

- If it is correctly echoed, receiver is behaving
- Performance or accuracy cost

Off-line testing like TBIT

- Blacklist misbehaving destination IP’s

ECN nonce protects all packets

# Backward compatibility

How do we deal with existing ECT/CE implementations?

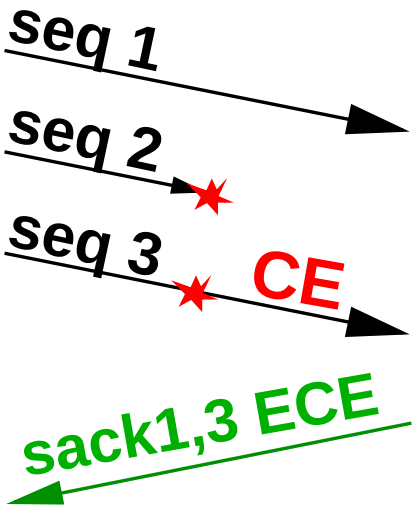
Routers: straightforward, described in latest ECN draft

TCPs:

- Use third bit in negotiation procedure? ($111 \rightarrow 101$)
- Easy to notice when nonce is unsupported? ($111 \rightarrow 001$)
- Support nonce-less ECN for transition period?

# SACKs

Should nonces cover SACK'd packets?




ECN-echo set for subsequent acks

Nonce applies only to in-order acknowledged segments

SACKs don't increase the window.

# Conclusion

ECN without nonces allows receivers to hide signals

- Hard to verify correct behavior
- Misbehaving receivers benefit

ECN with nonces prevents concealment

- One more header bit
- Minor TCP state

# Questions? Insults?

Doc: draft-ietf-tsvwg-tcp-nonce-00.txt

Offline: nspring@cs.washington.edu

Talk slides: `http://www.cs.washington.edu/homes/nspring/talks/ietf-ecn.ps.gz`